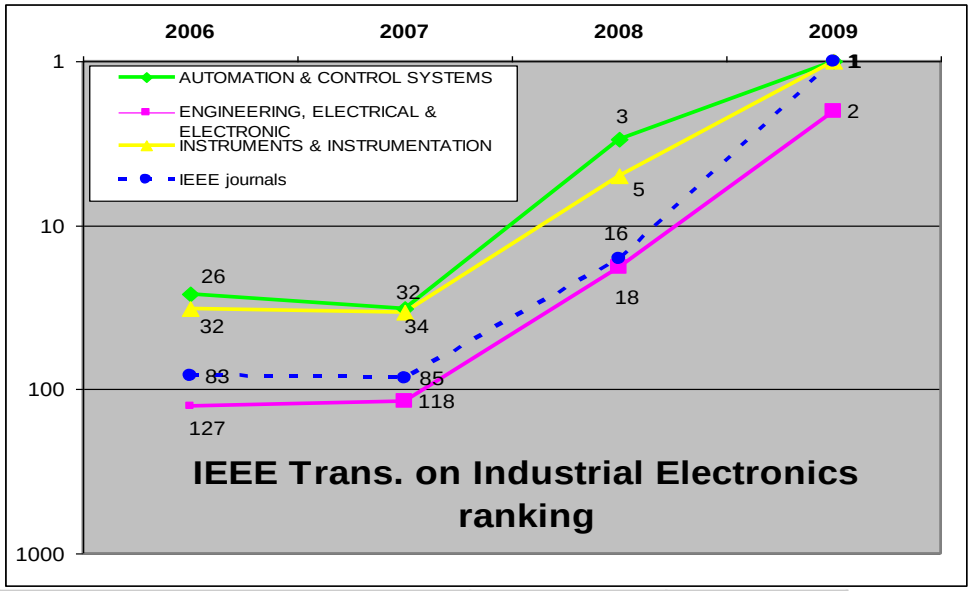
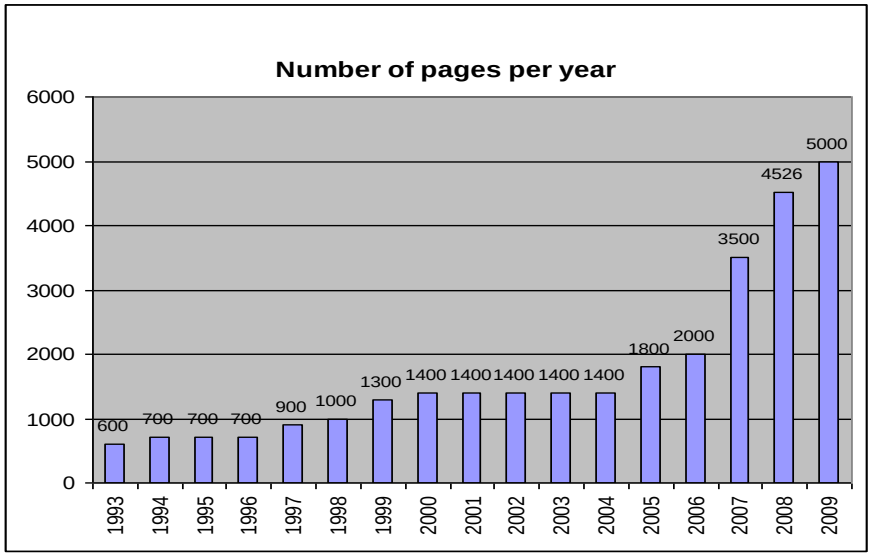
Human factor and computational intelligence limitations in resilient control systems

Prof. Bogdan M. Wilamowski, IEEE Fellow
Past Editor-in-Chief of IEEE Trans. on Industrial Electronics
Elected Editor-in-Chief of IEEE Trans. on Industrial Informatics
Director of Alabama Nano/Micro Science and Technology Center
Professor of Department of Electrical & Computer Engineering
200 Broun Hall, Auburn University, AL 36849-5201
ph. 334-844-1629 fax 334-844-1888 home 334-501-9092
tieedit@auburn.edu *http://www.eng.auburn.edu/~wilambm/*

IEEE Transactions on Industrial Electronics

http://tie.ieee-ies.org/tie/ **IF=5.468**





Category Name	Total Journals in Category	Journal Rank in Category
AUTOMATION & CONTROL SYSTEMS	53	1
ENGINEERING, ELECTRICAL & ELECTRONIC	229	2
INSTRUMENTS & INSTRUMENTATION	56	1
IEEE journals	96	1

IEEE USAGE OF MANUSCRIPT CENTRAL

- Number of Sites - 128
- 2008 Submissions – 46,542
- IEEE sites with most submissions
  - Transactions on Industrial Electronics
  - Communications Letters
  - Transactions on Signal Processing
  - Transactions on Wireless Communications
  - Photonics Technology Letters
  - Electron Device Letters

THOMSON REUTERS

IEEE-INST ELECTRICAL ELECTRONICS ENGINEERS INC

Journals 1 - 20 (of 96)

[1 | 2 | 3 | 4 | 5]

MARK ALL | UPDATE MARKED LIST

Ranking is based on your journal and sort sele

Mark	Rank	Abbreviated Journal Title (linked to journal information)	ISSN	JCR Data			
				Total Cites	Impact Factor	5-Year Impact Factor	Immediacy Index
☐	1	IEEE T IND ELECTRON	0278-0046	9014	5.468	4.665	0.460
☐	2	P IEEE	0018-9219	17993	4.613	6.824	0.566
☐	3	IEEE J SEL AREA COMM	0733-8716	13838	4.249	5.615	0.361
☐	4	IEEE T MED IMAGING	0278-0062	10426	4.004	5.544	0.468
☐	5	IEEE T INFORM THEORY	0018-9448	29333	3.793	5.434	0.420
☐	6	IEEE SIGNAL PROC MAG	1053-5888	3040	3.758	6.157	0.733
☐	7	IEEE T EVOLUT COMPUT	1089-778X	3849	3.736	6.612	0.469
☐	8	IEEE T NEURAL NETWOR	1045-9227	9883	3.726	4.144	0.330
☐	9	IEEE T FUZZY SYST	1063-6706	5211	3.624	4.804	0.266
☐	10	IEEE T POWER ELECTR	0885-8993	7719	3.483	3.813	0.405
☐	11	IEEE J SOLID-ST CIRC	0018-9200	13137	3.466	4.037	0.295

General remarks

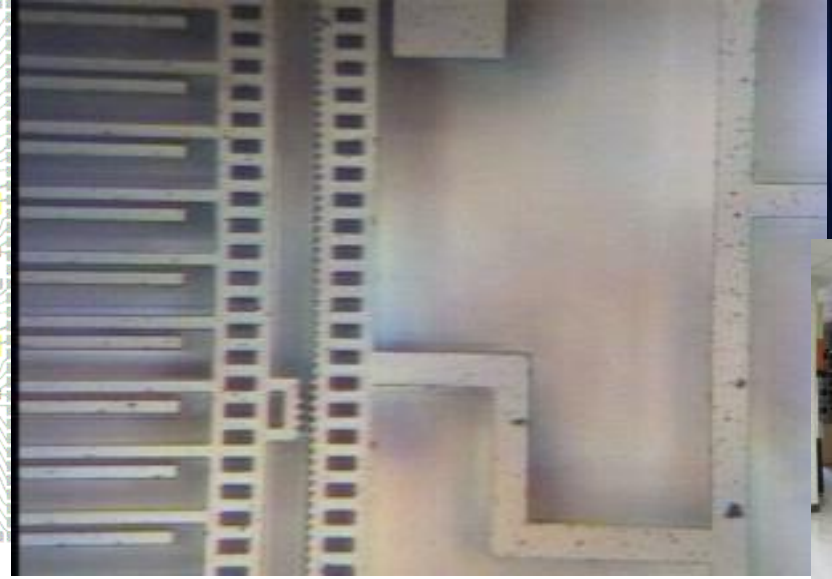
- **citations are becoming more important than publications**
- **journal paper ≠ journal paper**

How to get cited?

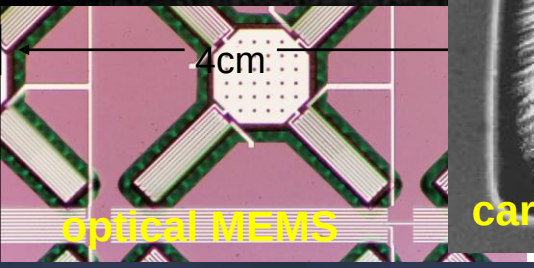
- Current content (references)
- Clear description of the idea
- Verification of results
- Wording of the title
- Abstract

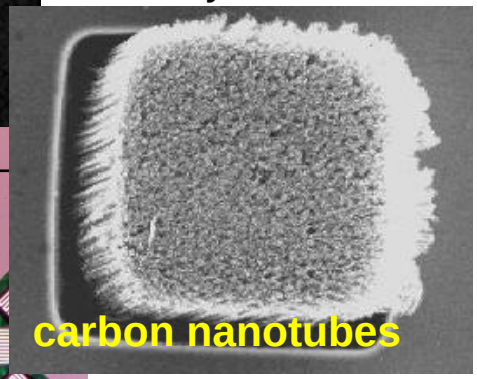
Alabama Micro Nano Science and Technology Center

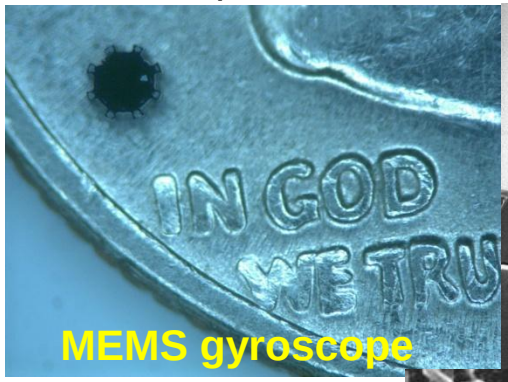
AUBURN

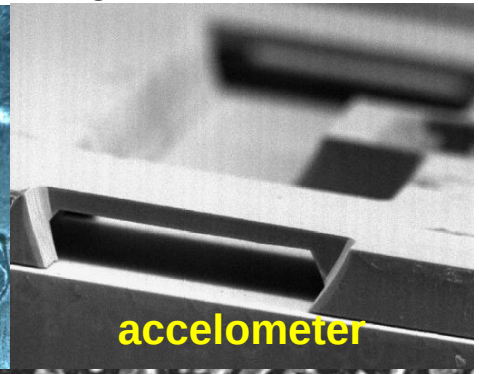


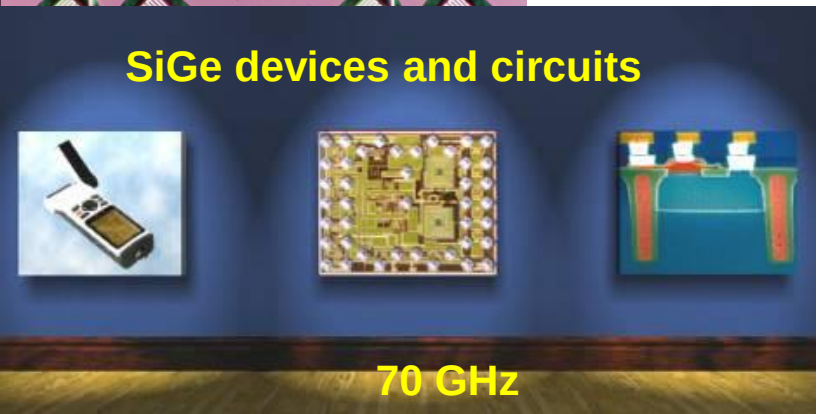
30 faculty members, 4 staff, 3 co-op students, and 40 graduate students.

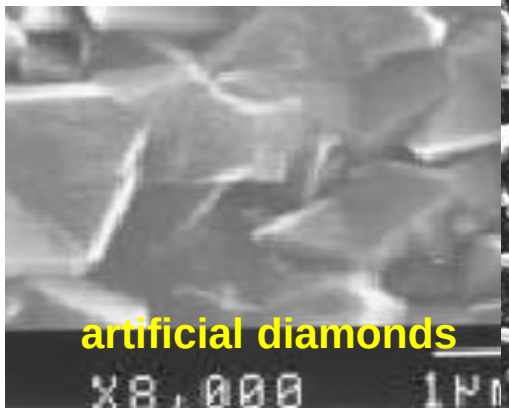


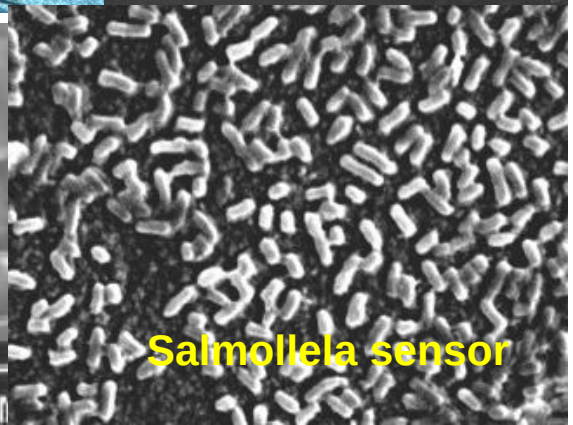










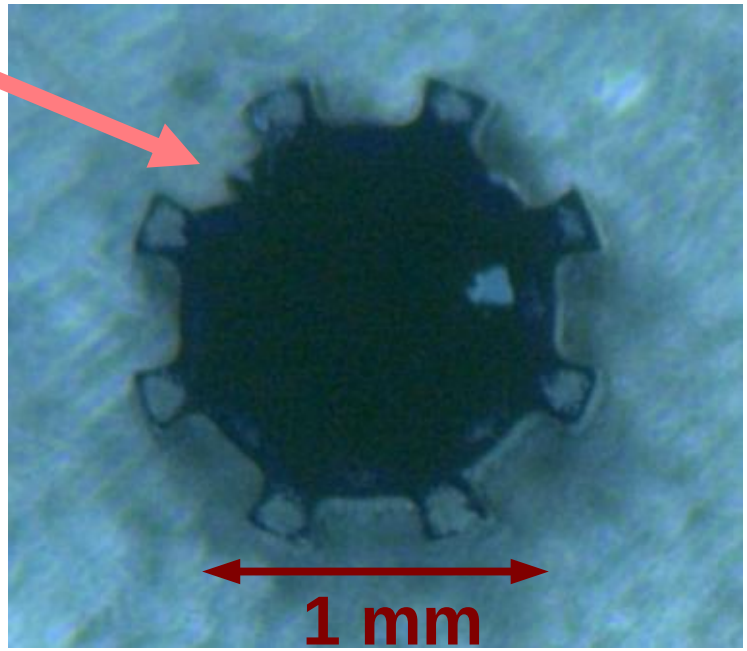


Alabama Micro Nano Science and Technology Center

AUBURN

MEMS Annular Rotating Sensor (MARS)

MEMS - gyroscope



All fabrication processes done in the ANMSTC

Humans are relatively unreliable element in control systems

Humans are capable not only of solving many scientific problems, but also in the process they are making mistakes. For example, even very good students are not able to solve long problems requiring complex calculations without making errors in the process

In the early stage of development of VLSI chips, required mask was manually cut on a foil and when the number of transistors in the circuits was larger than 50, then chances of successful production of a chip was very slim.

Also, with usage of computers humans were the weakest link in the chip design process where many different software were used. For example, when humans were required to manually enter data obtained from one software to another software then again errors were generated.

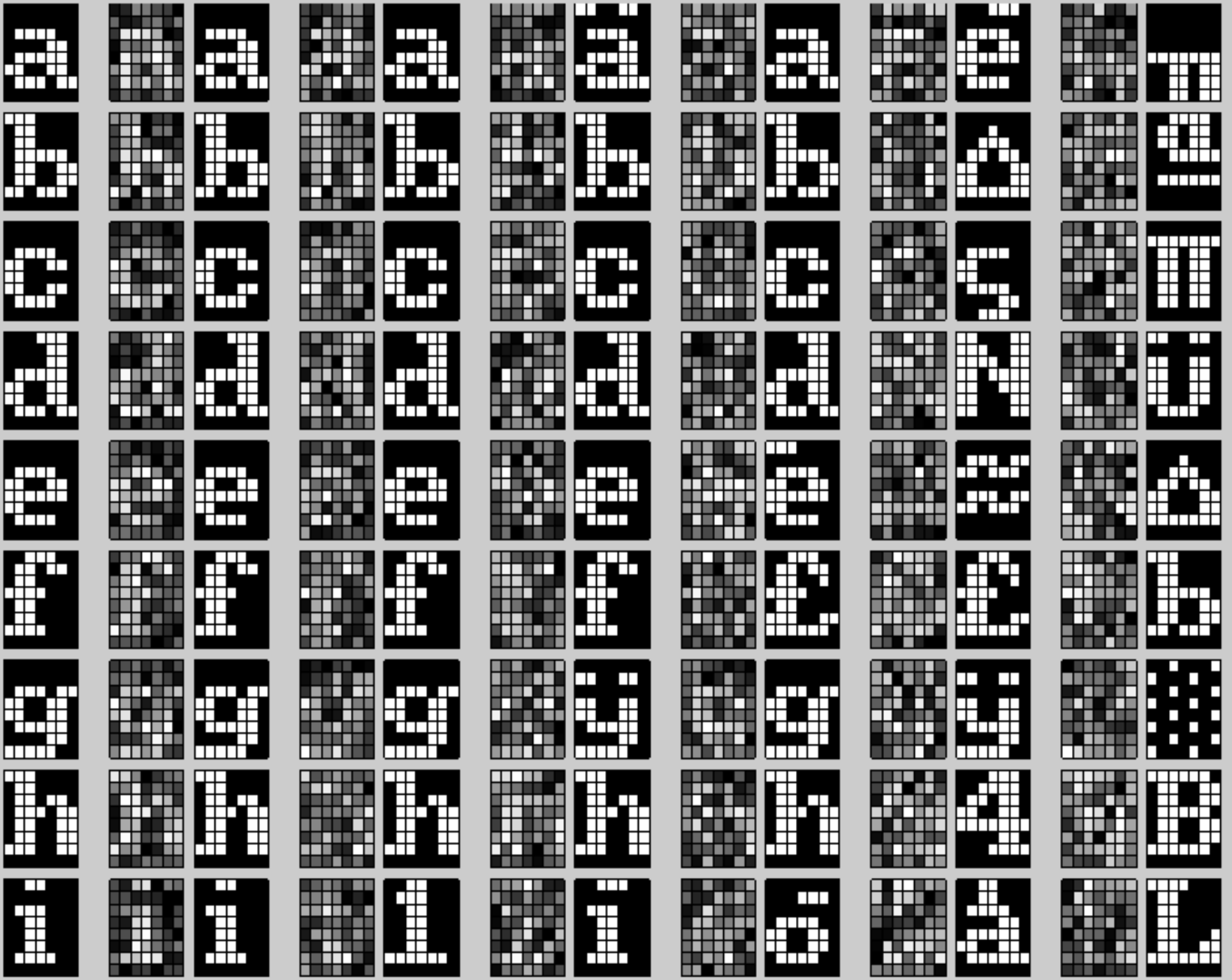
Conclusion: We should try to replace humans in control systems if possible

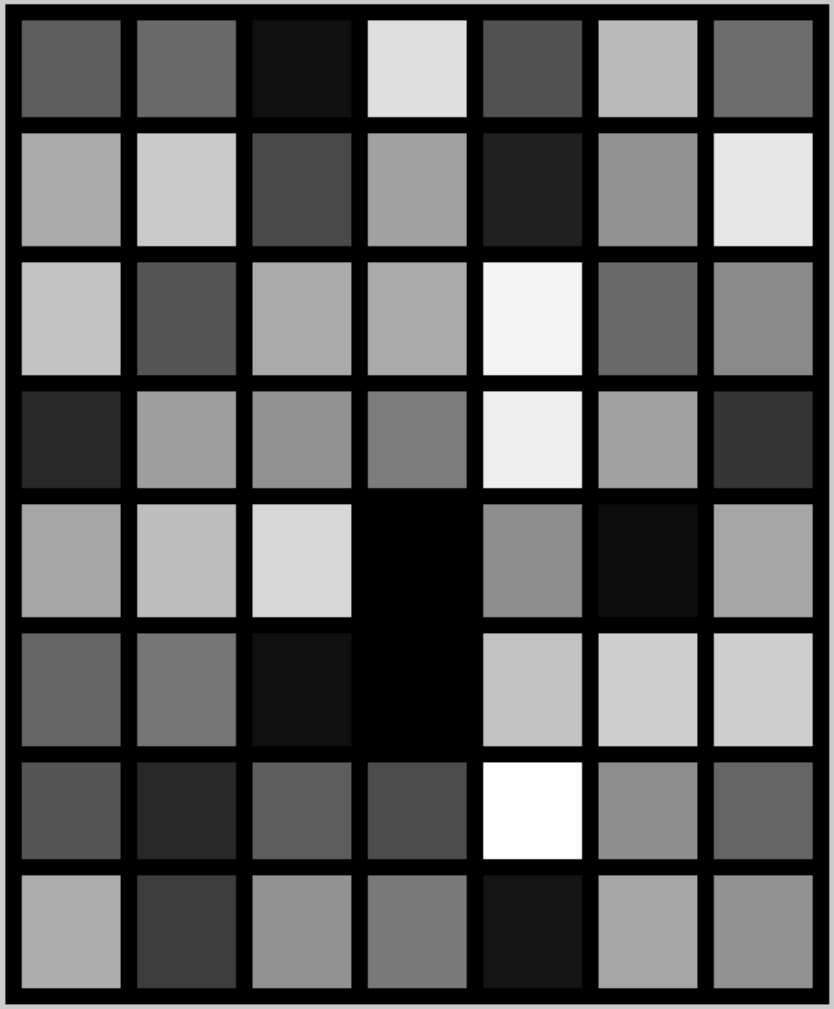
Methods of artificial / computational intelligences are possibly best candidates to replace humans

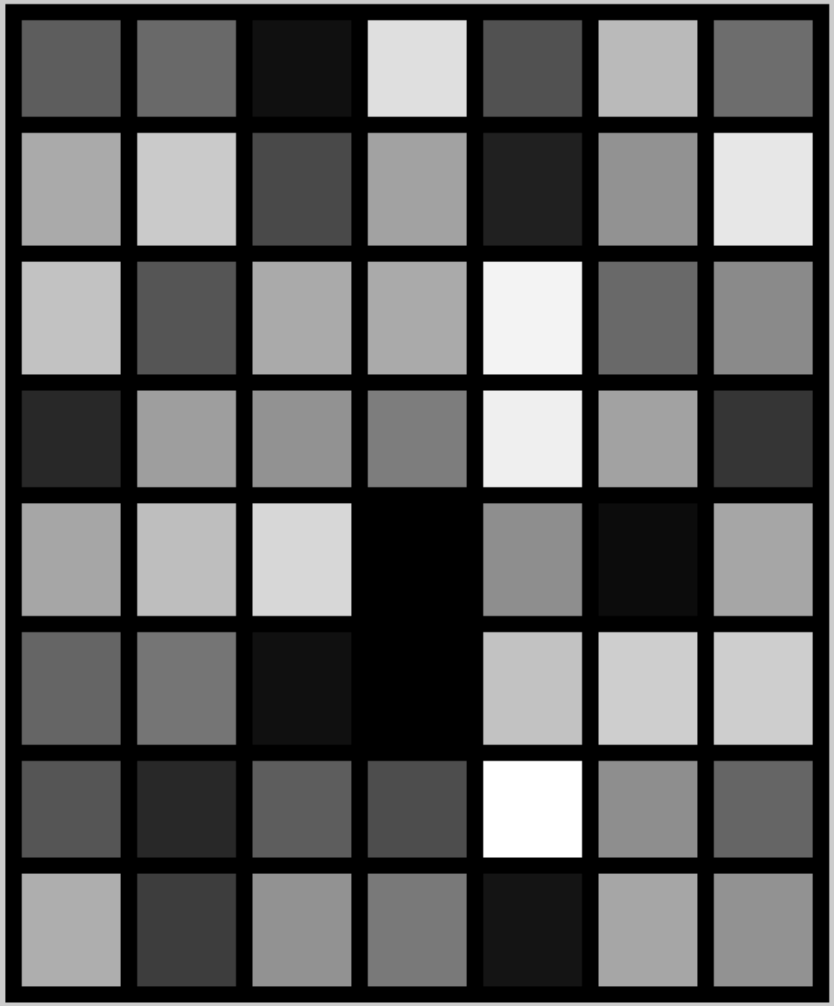
What is the difference between artificial intelligence and computational intelligence?

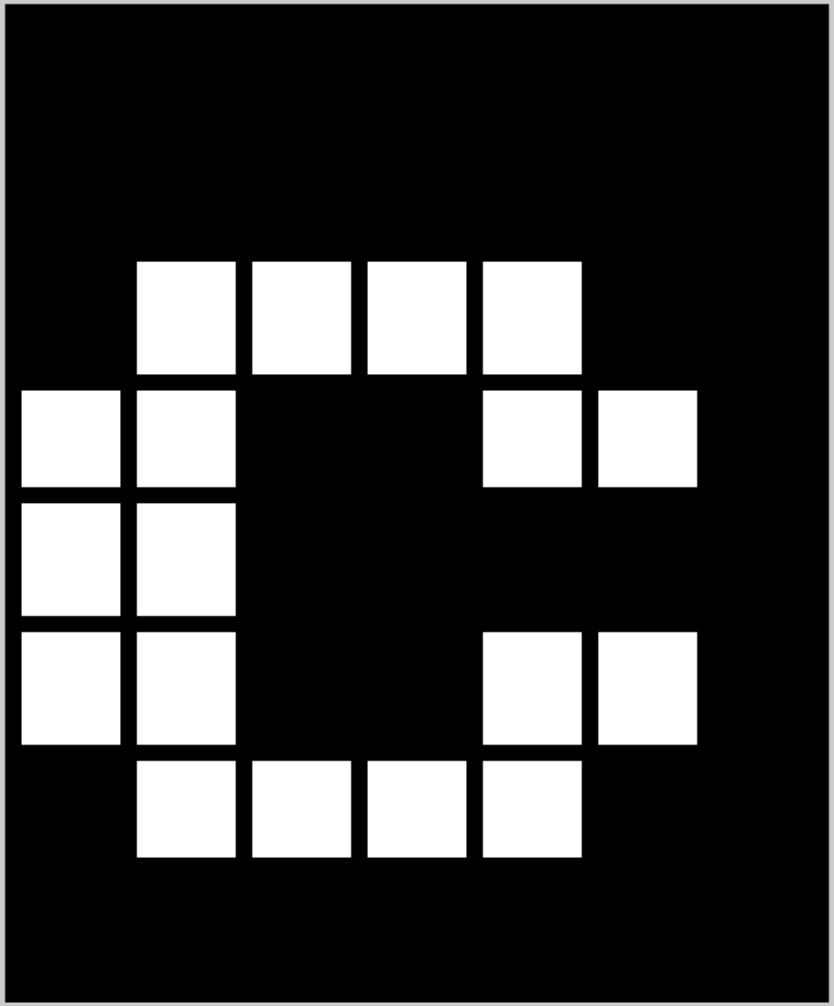
The goal of **artificial intelligence** is to be undistinguishable form humans.

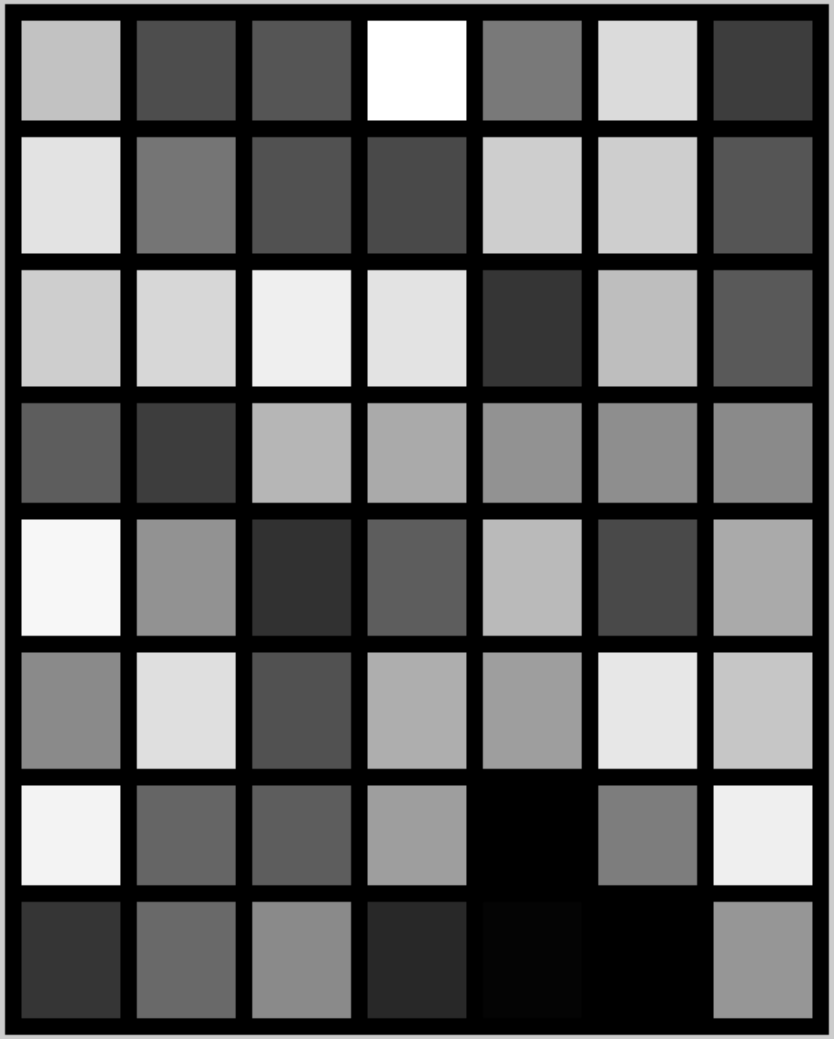
Computational intelligence is trying to solve problems better than humans

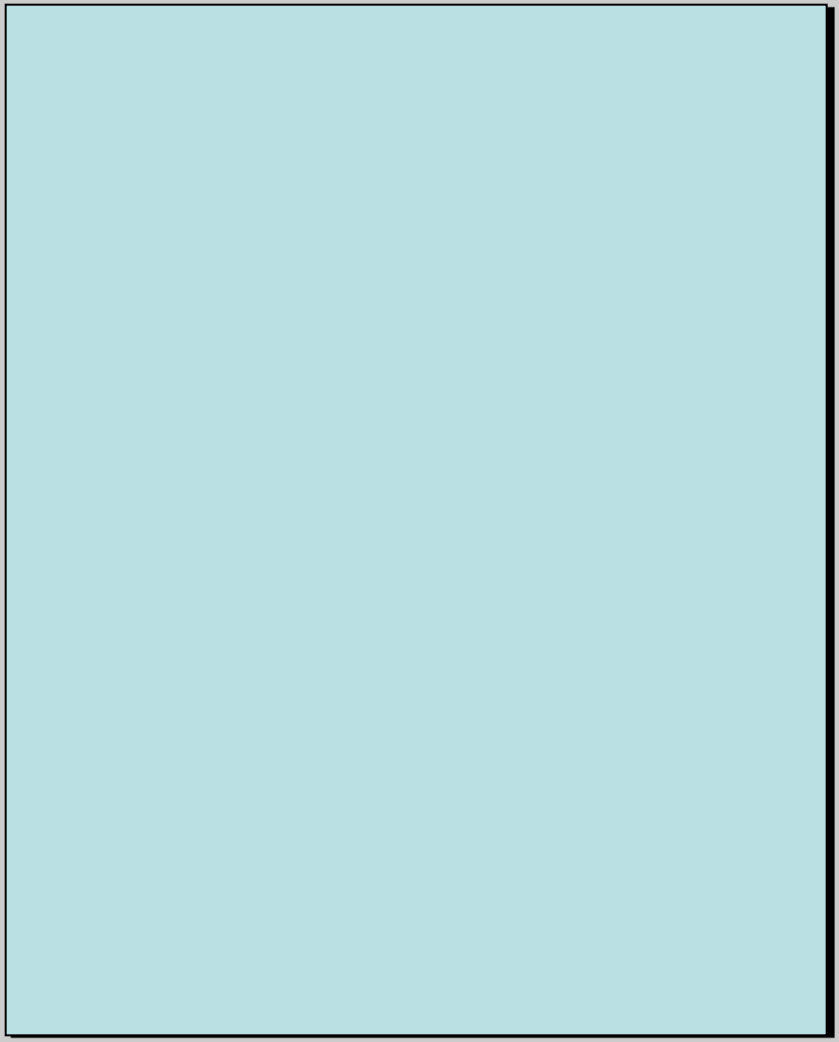


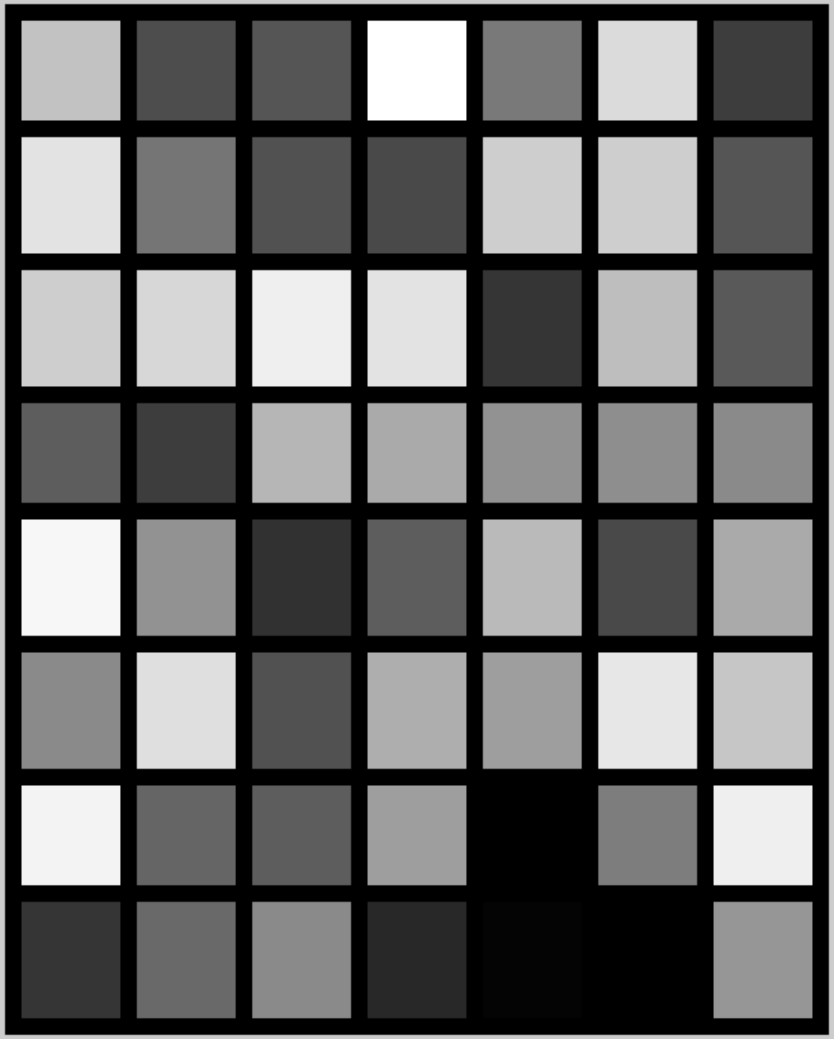


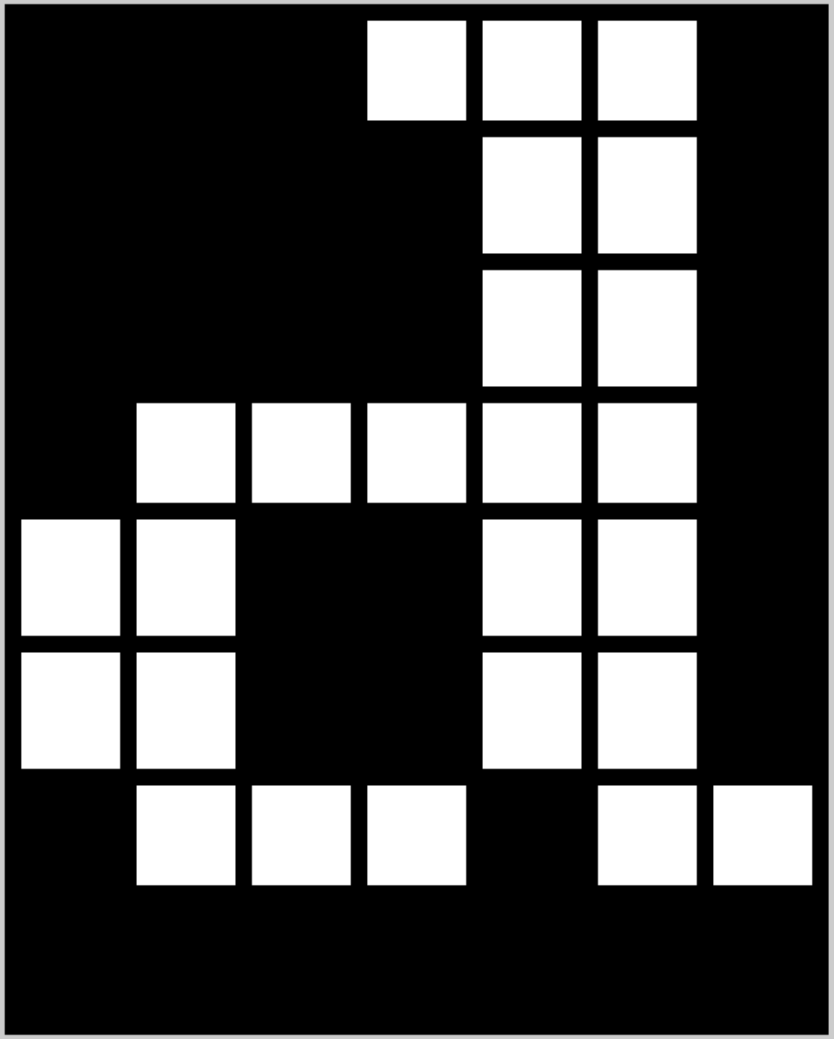


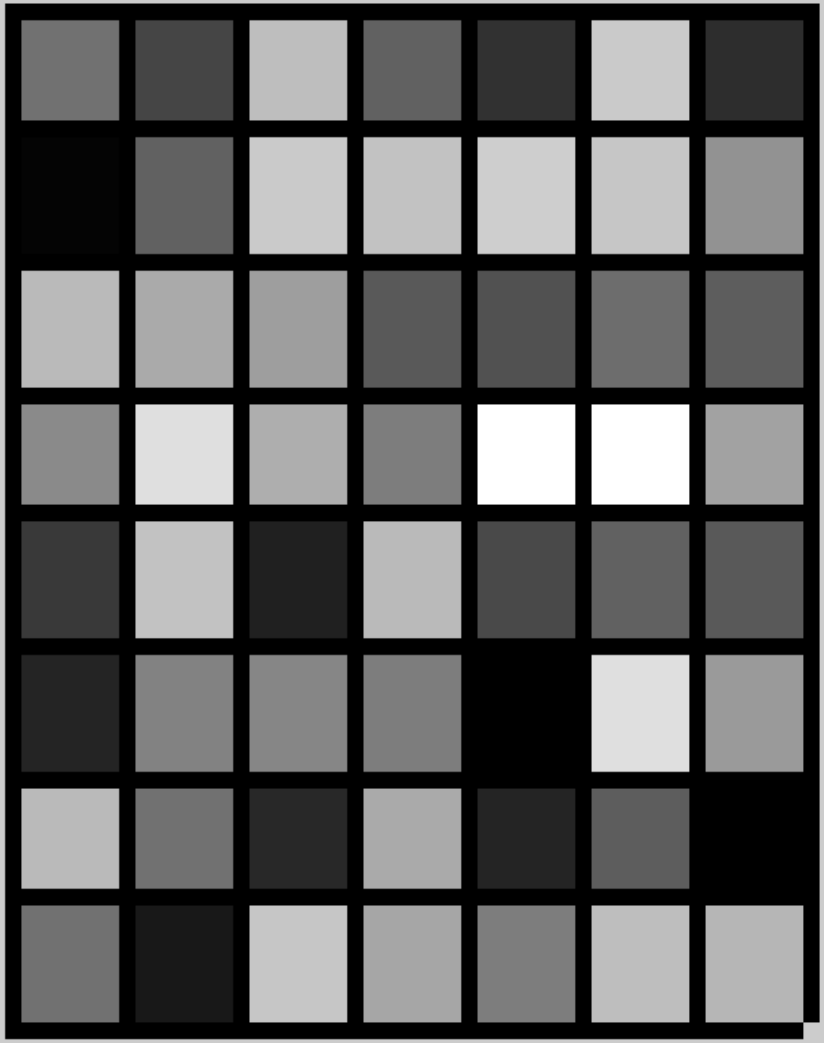


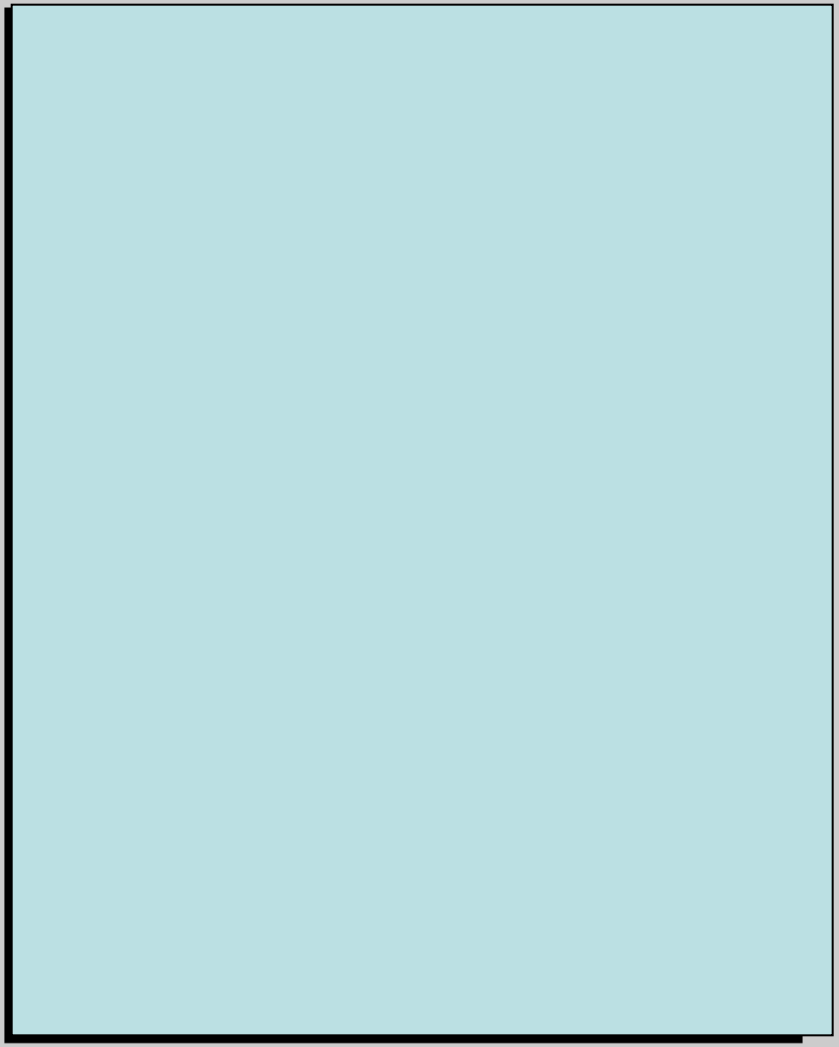


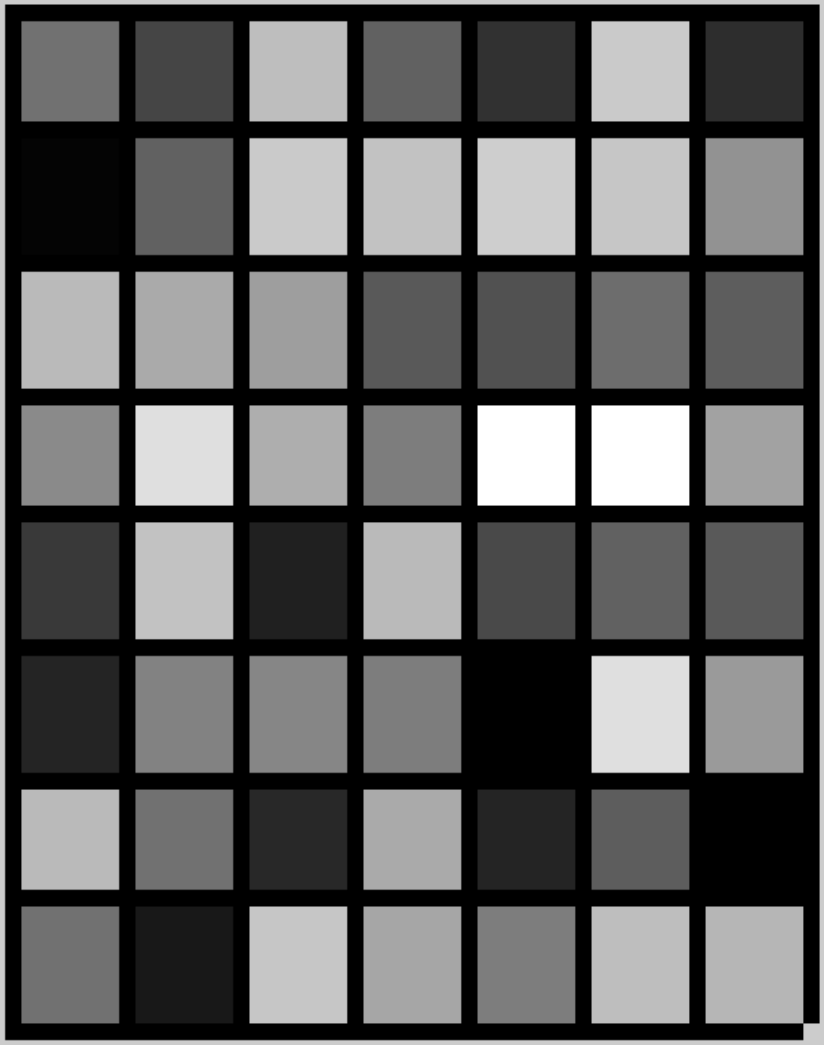


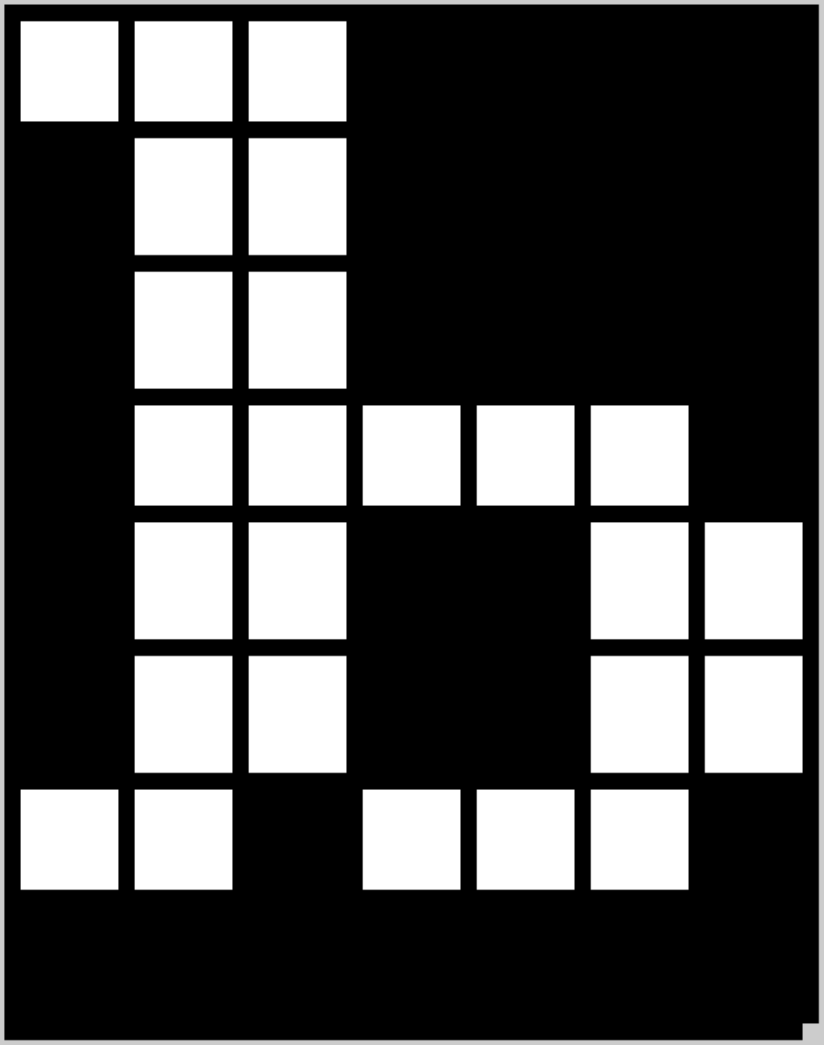


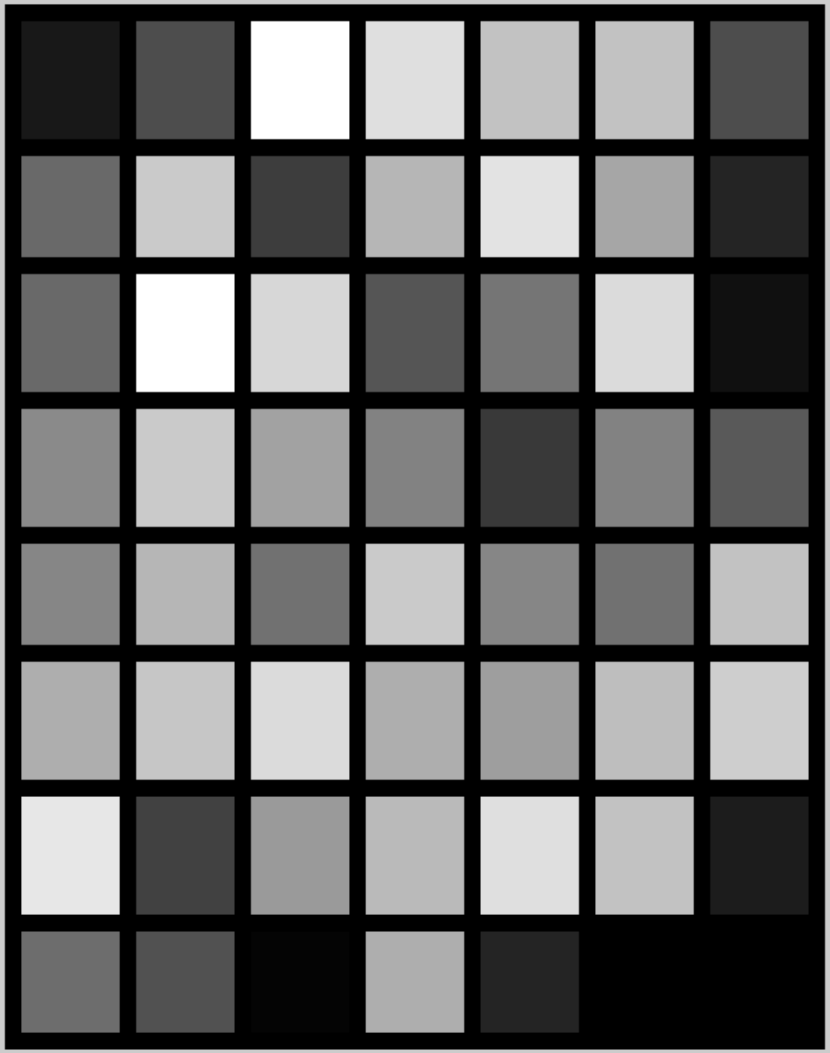


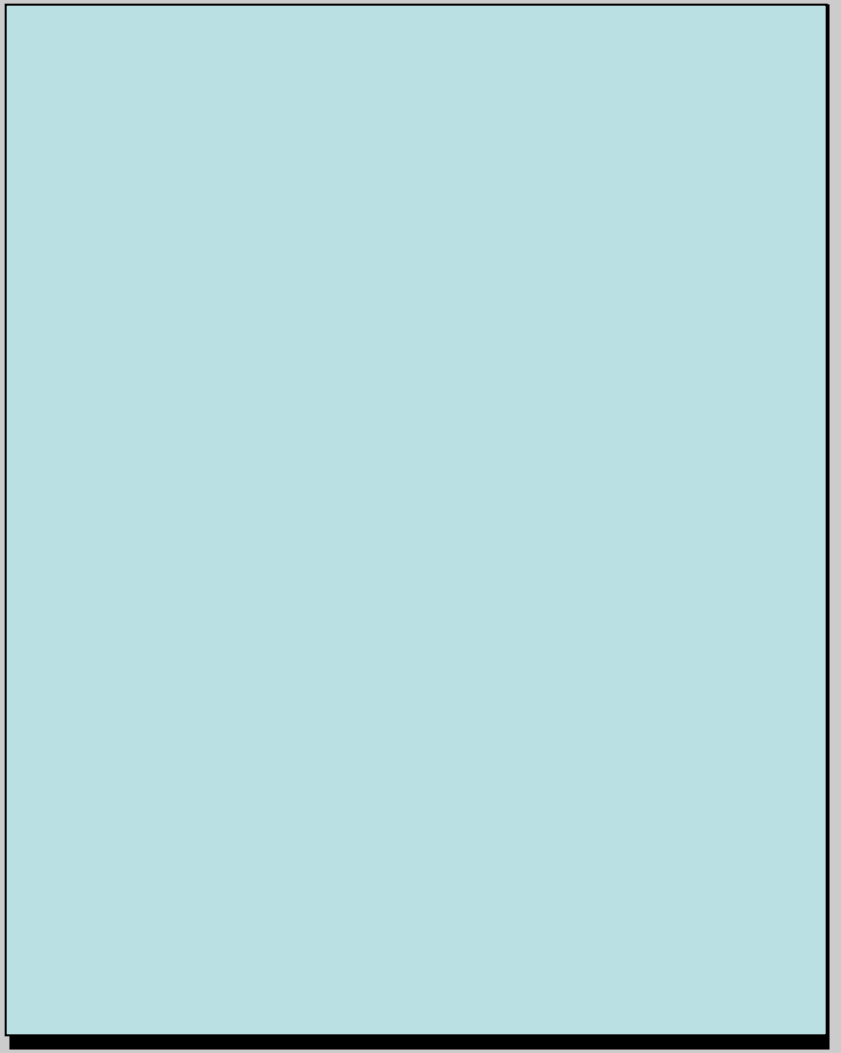


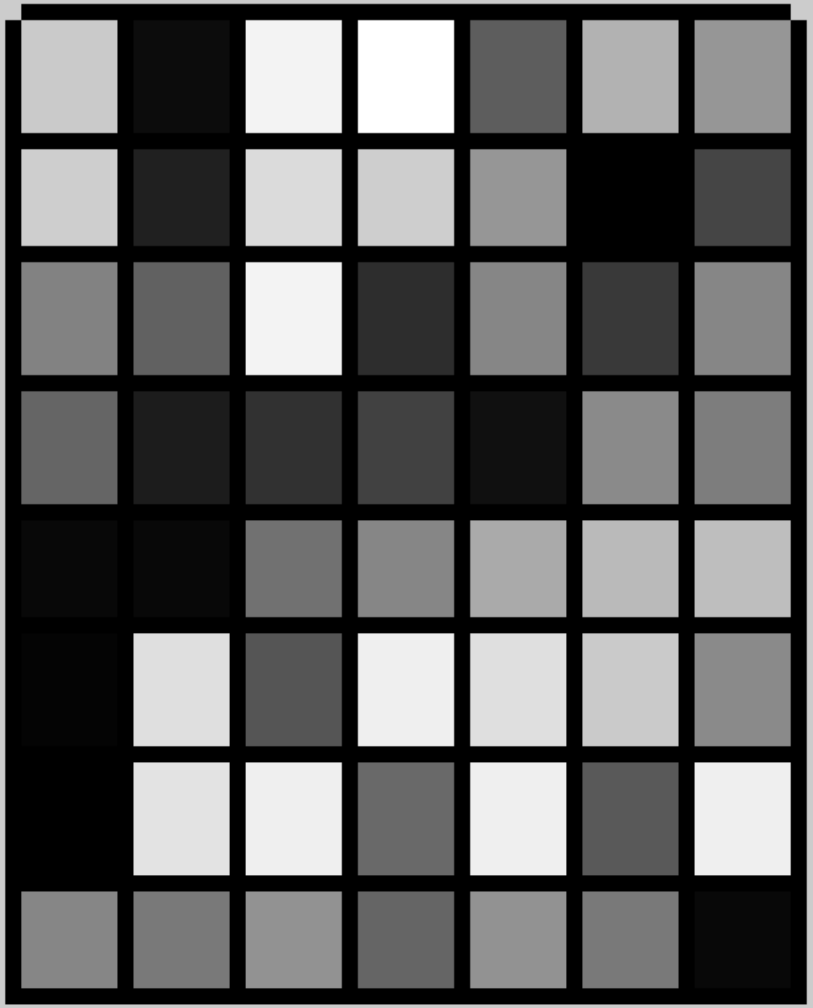


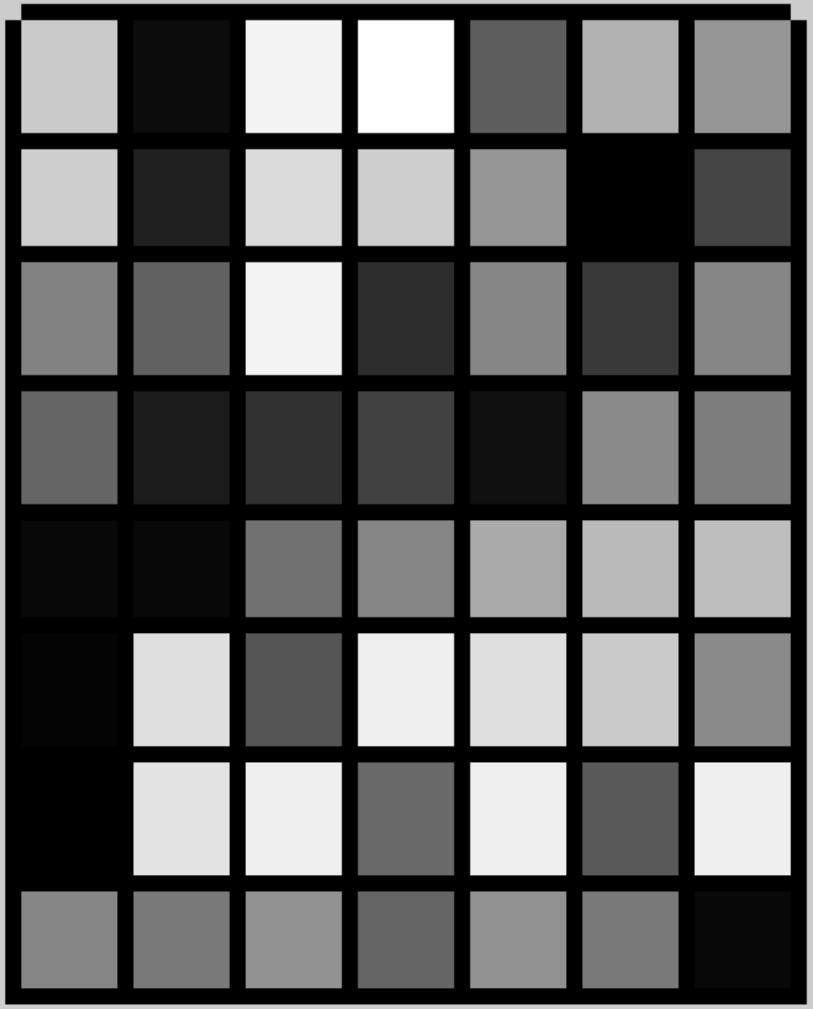


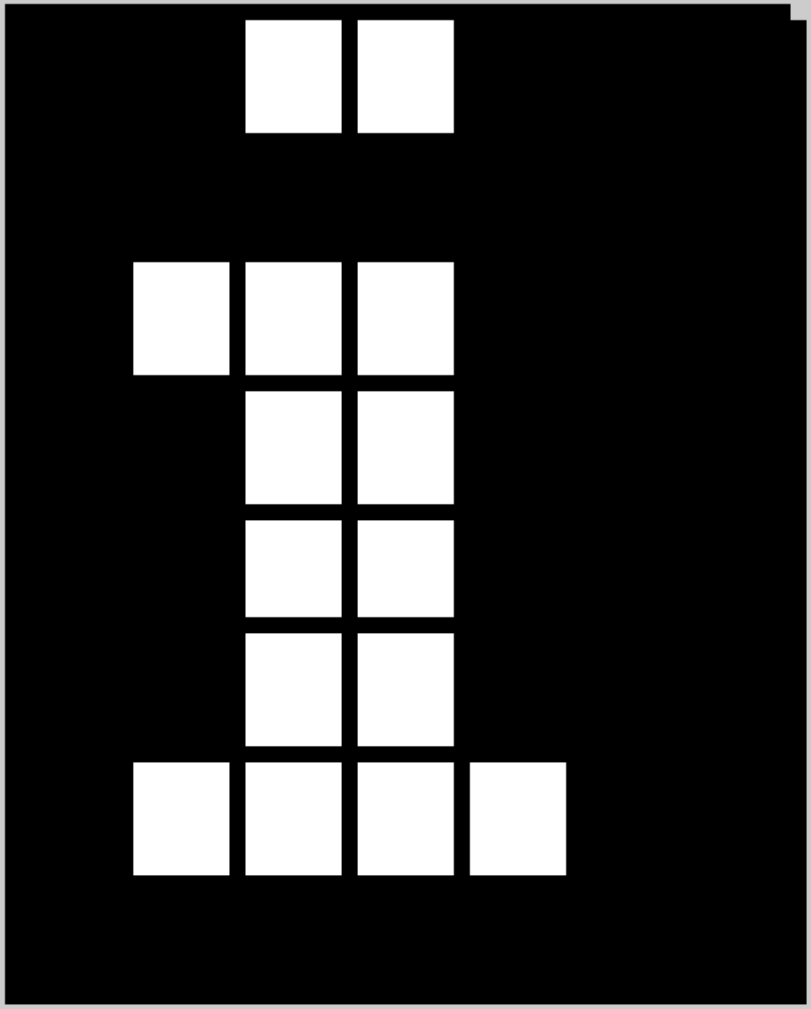


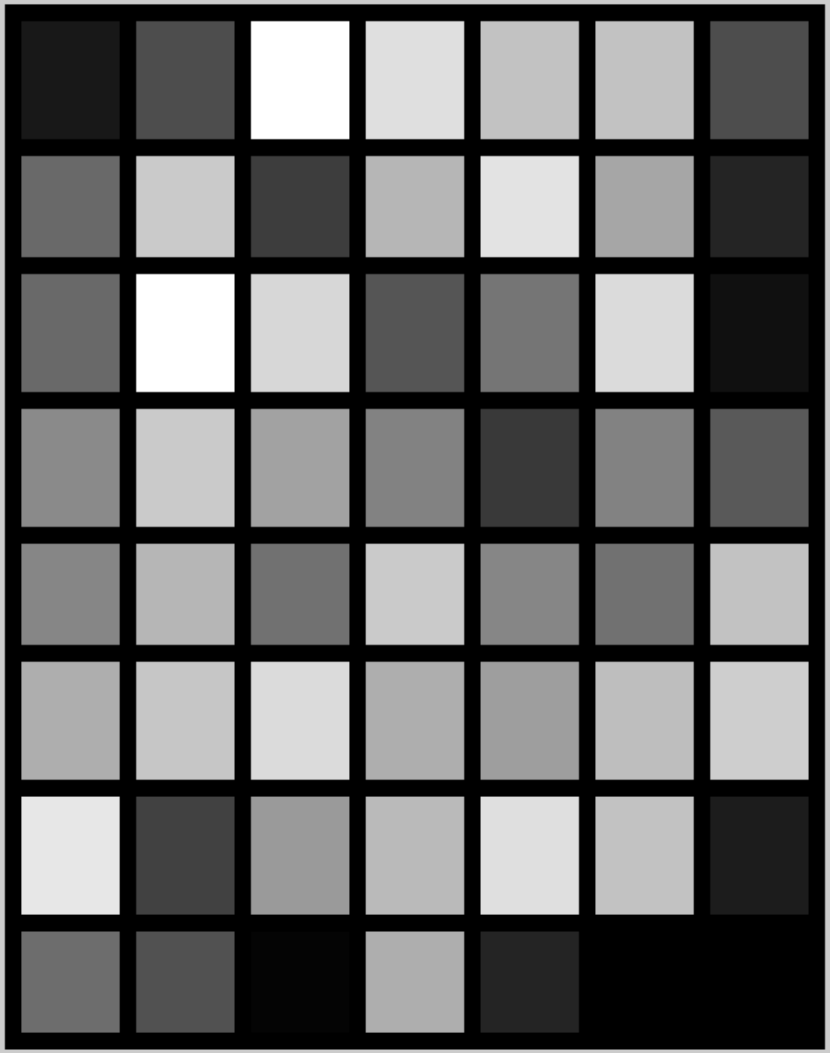


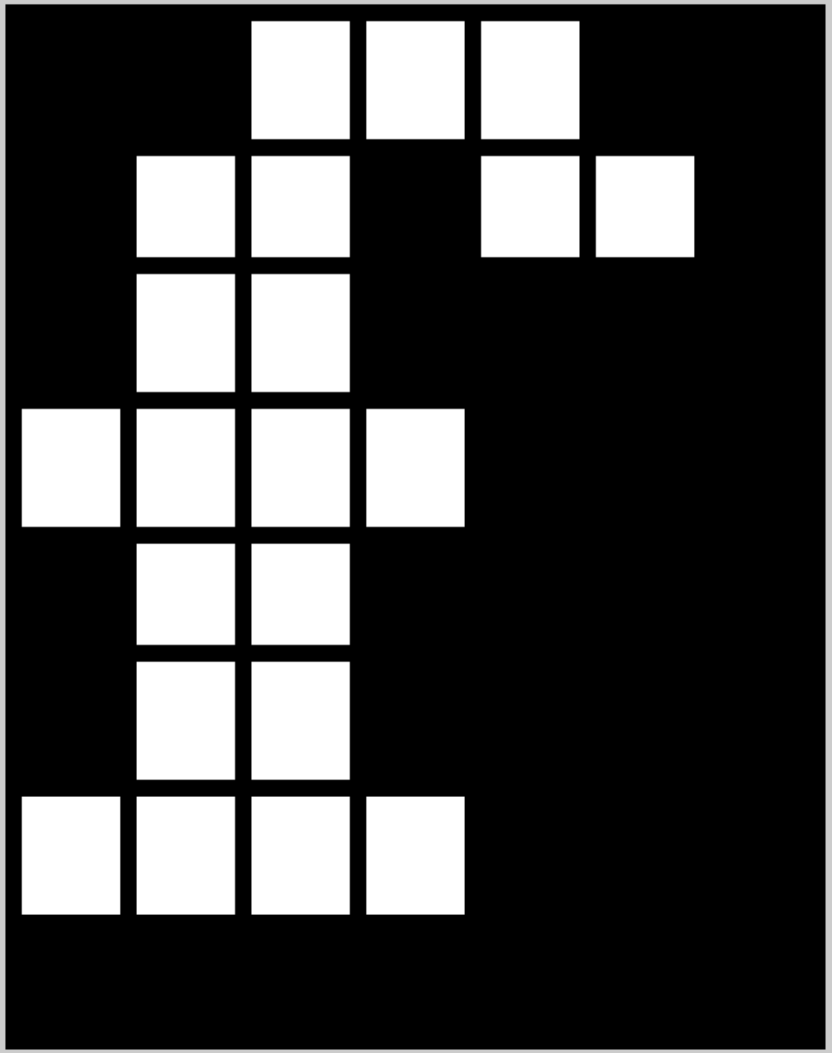


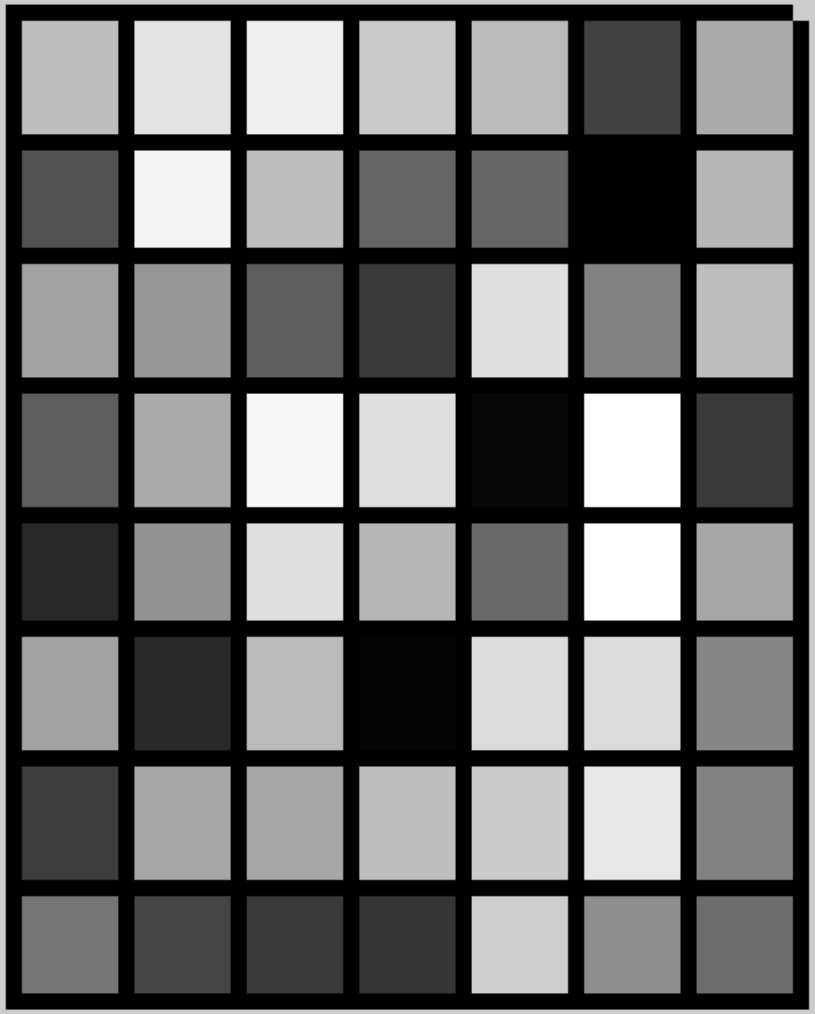


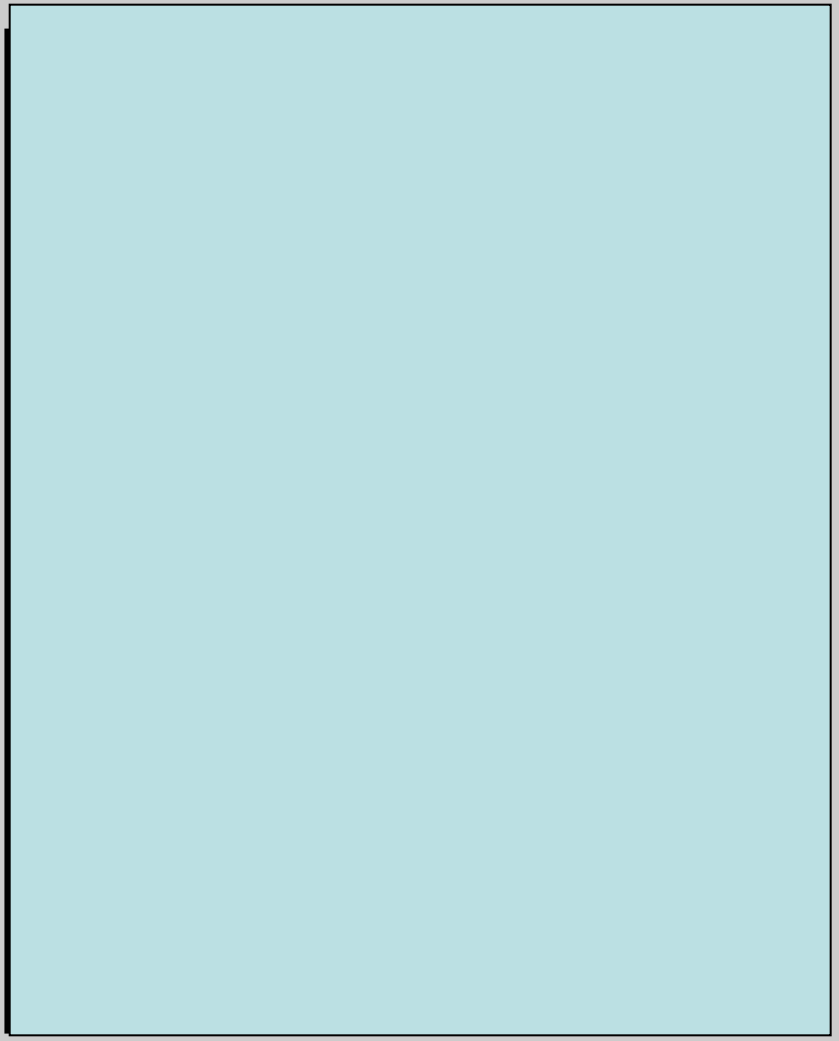


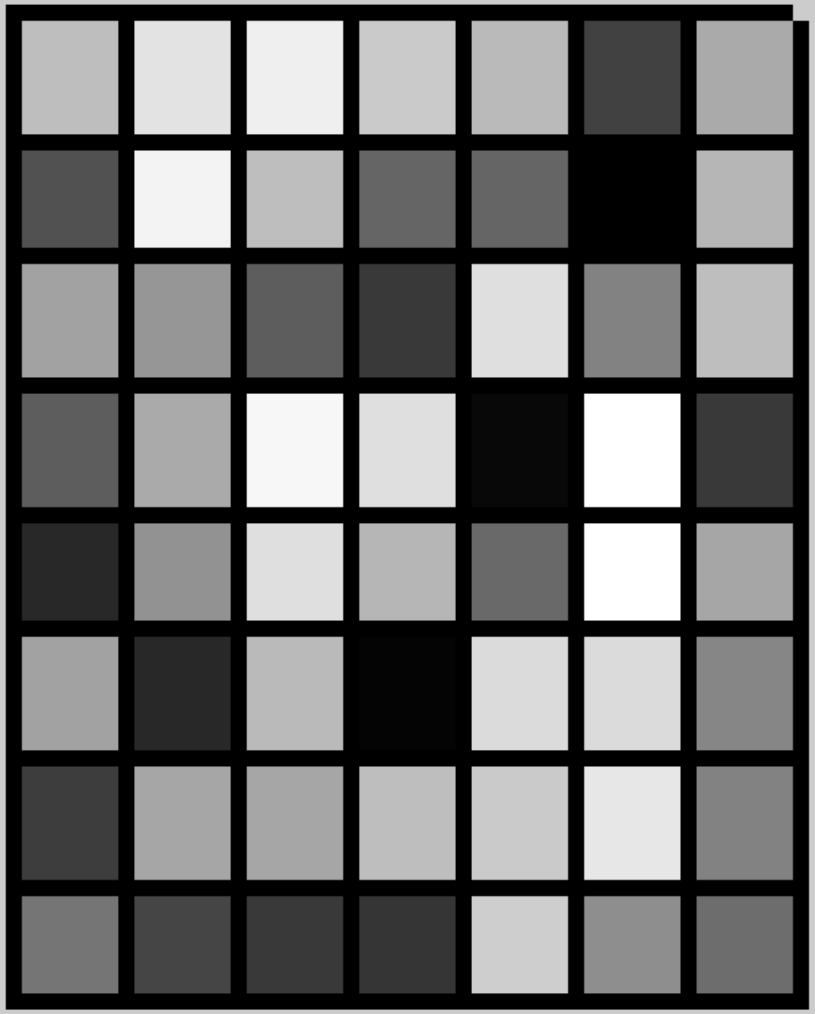


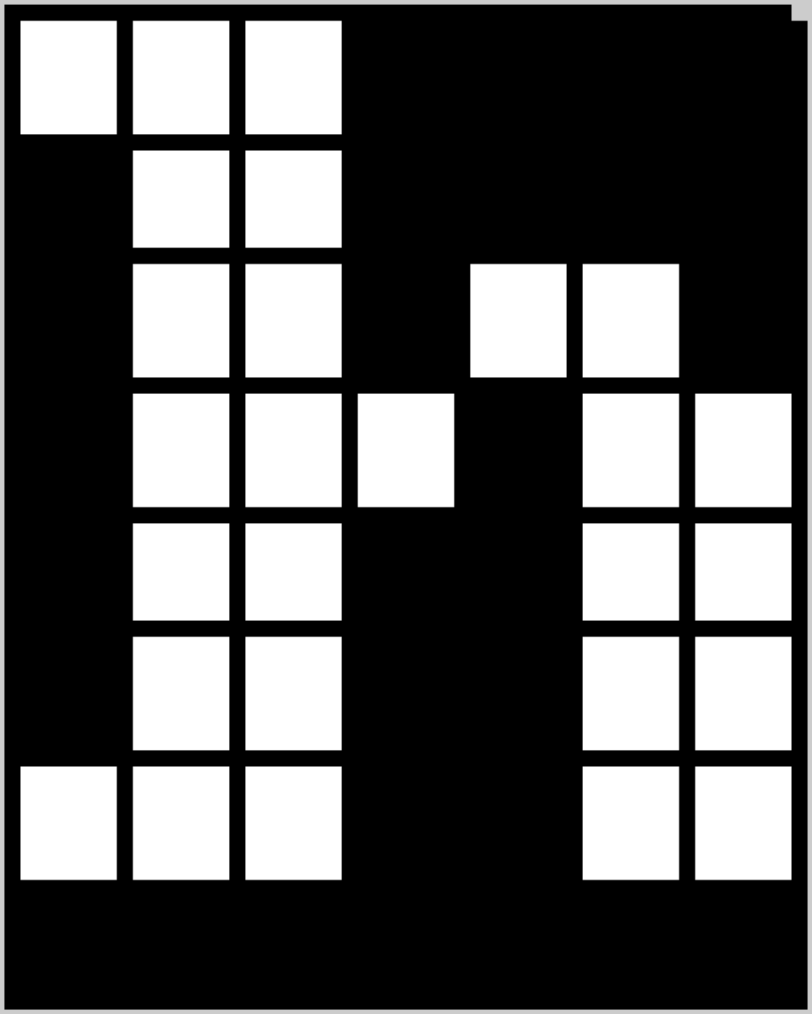


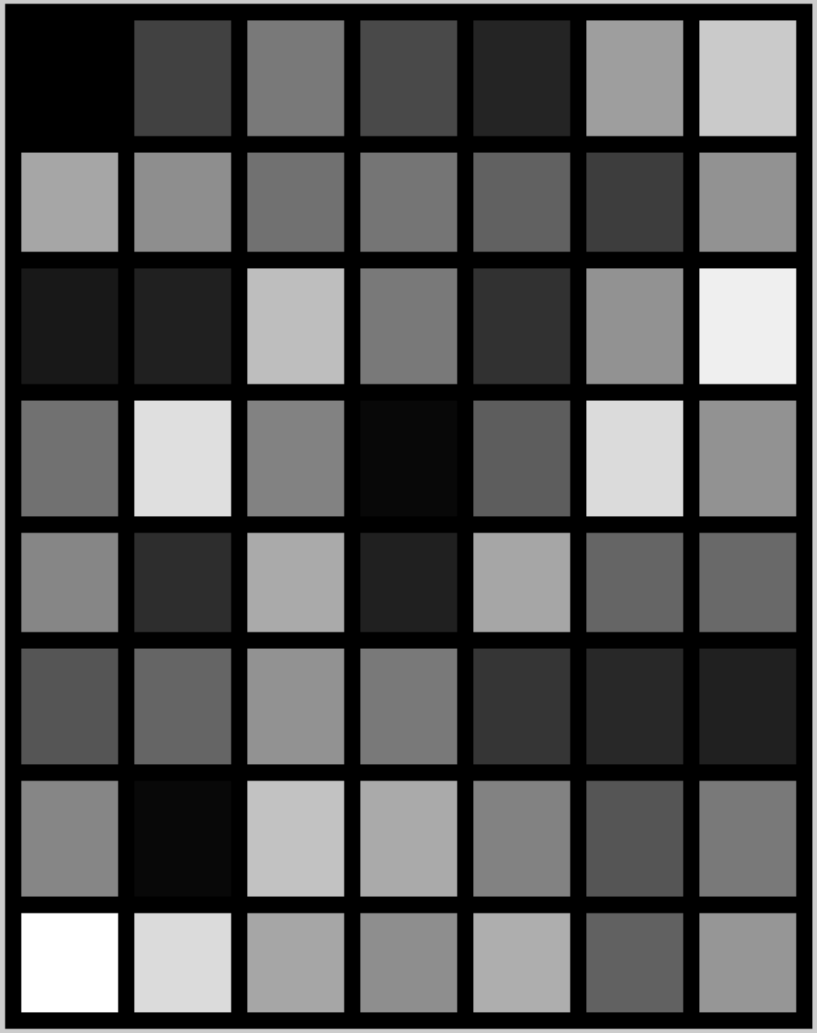


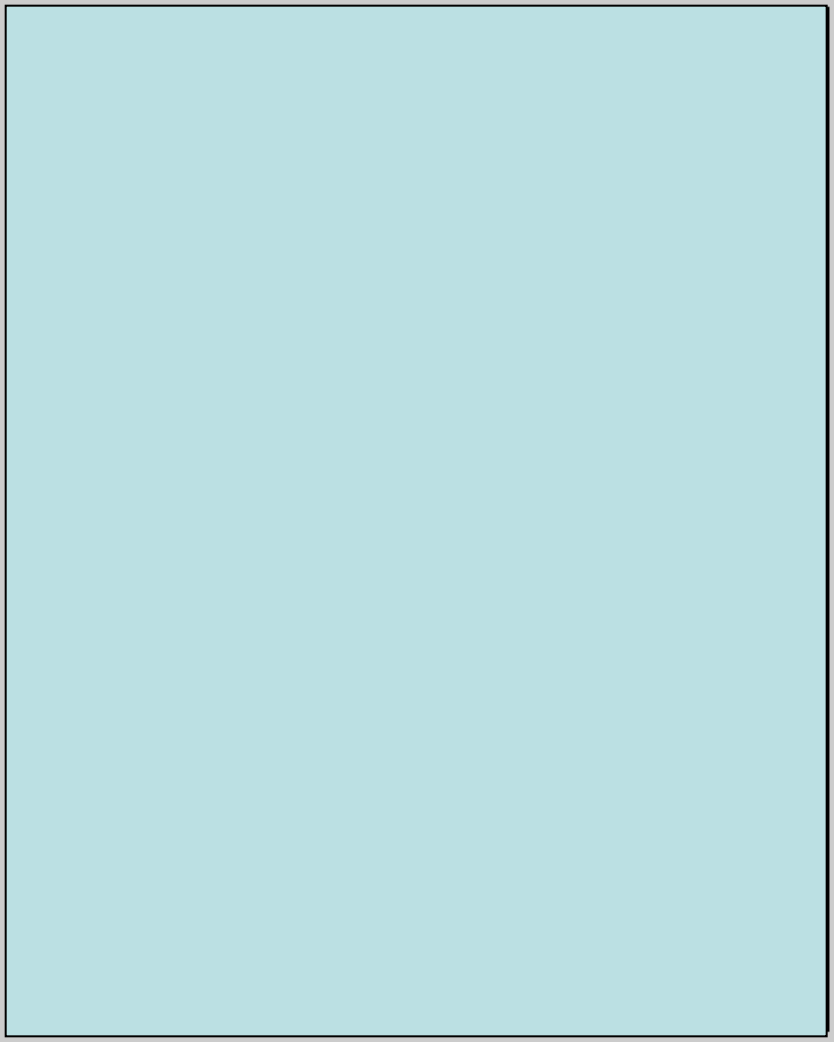


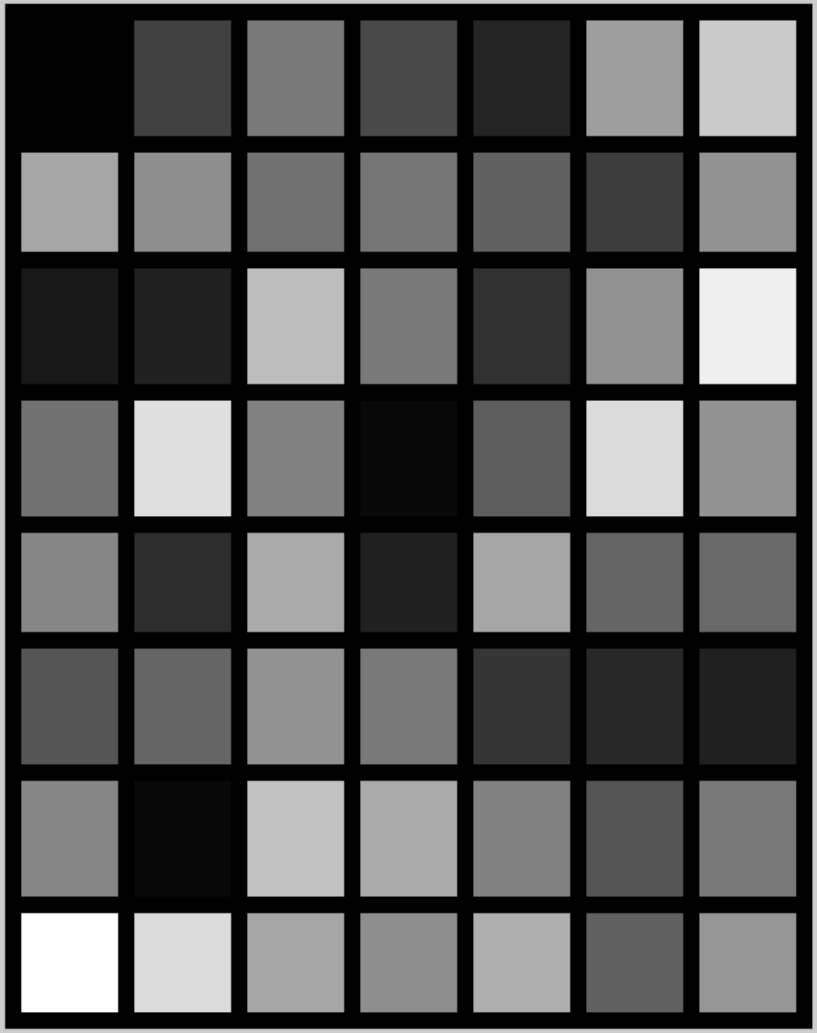


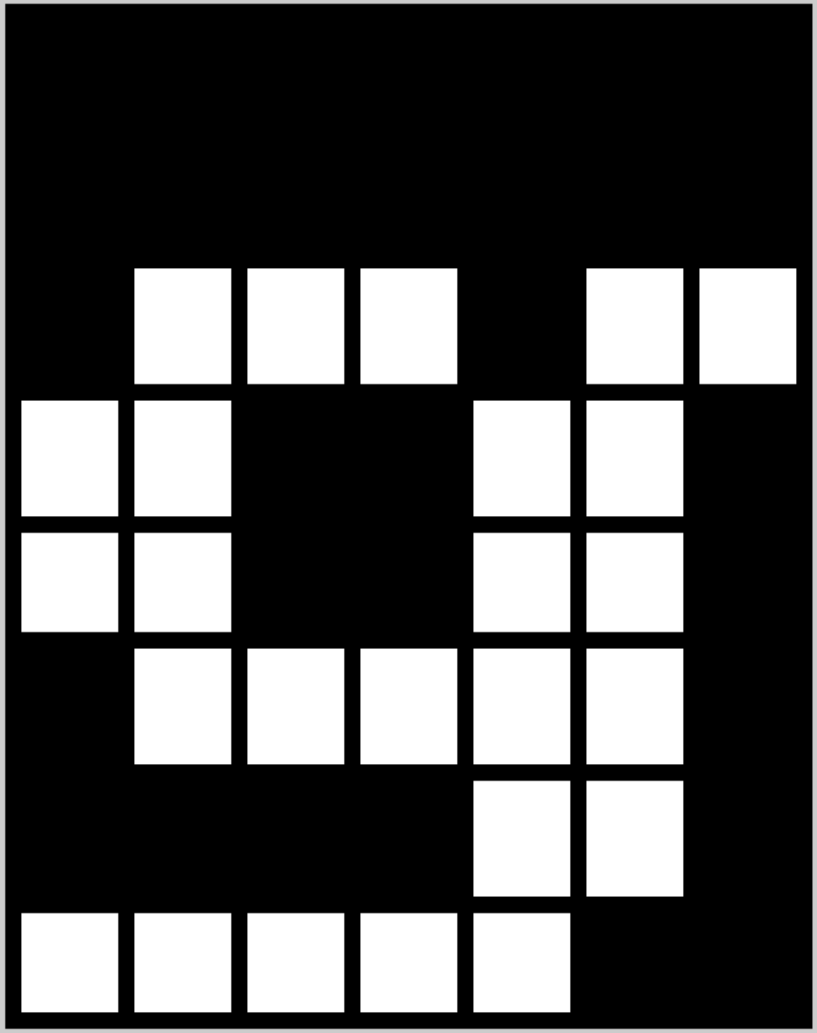


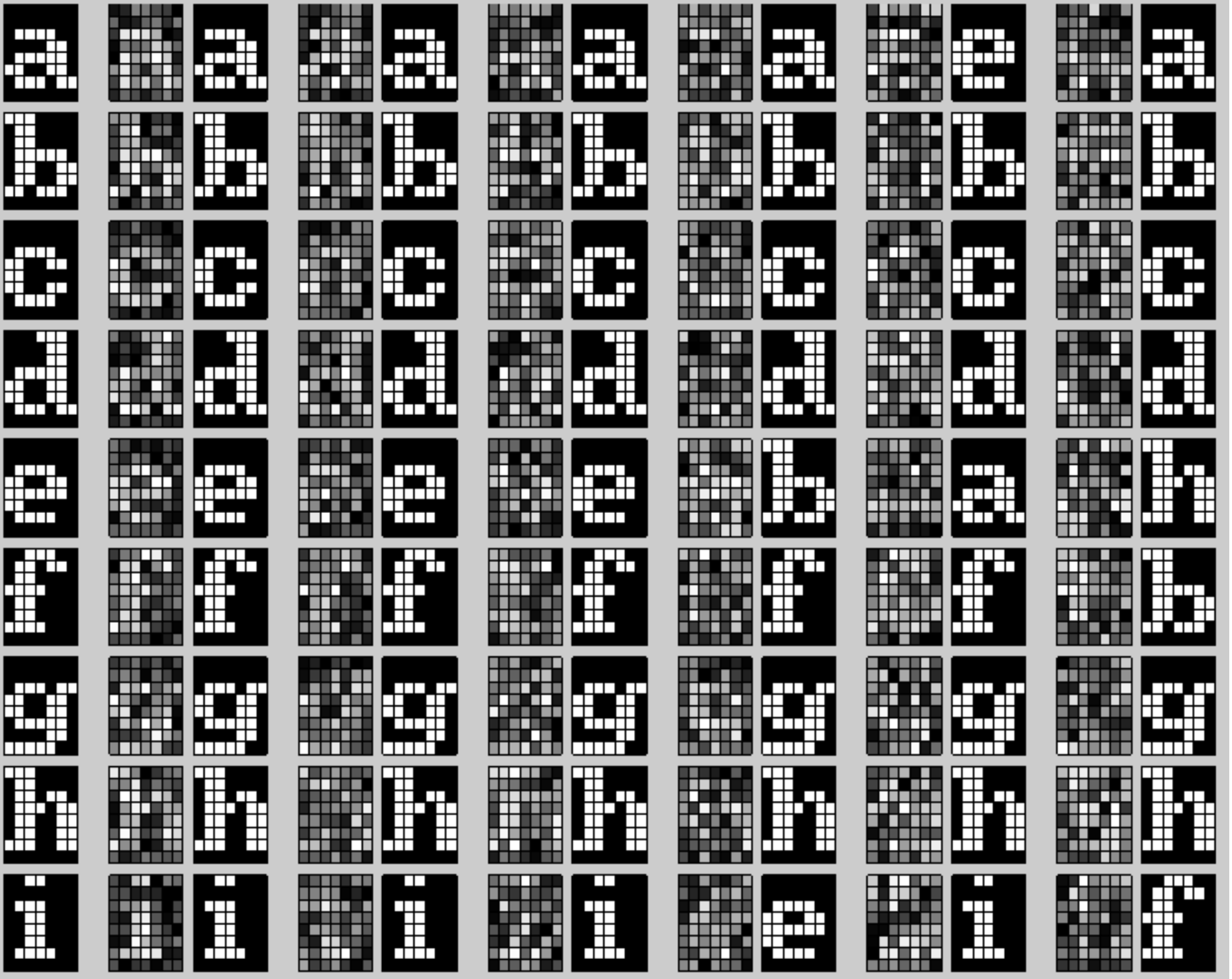


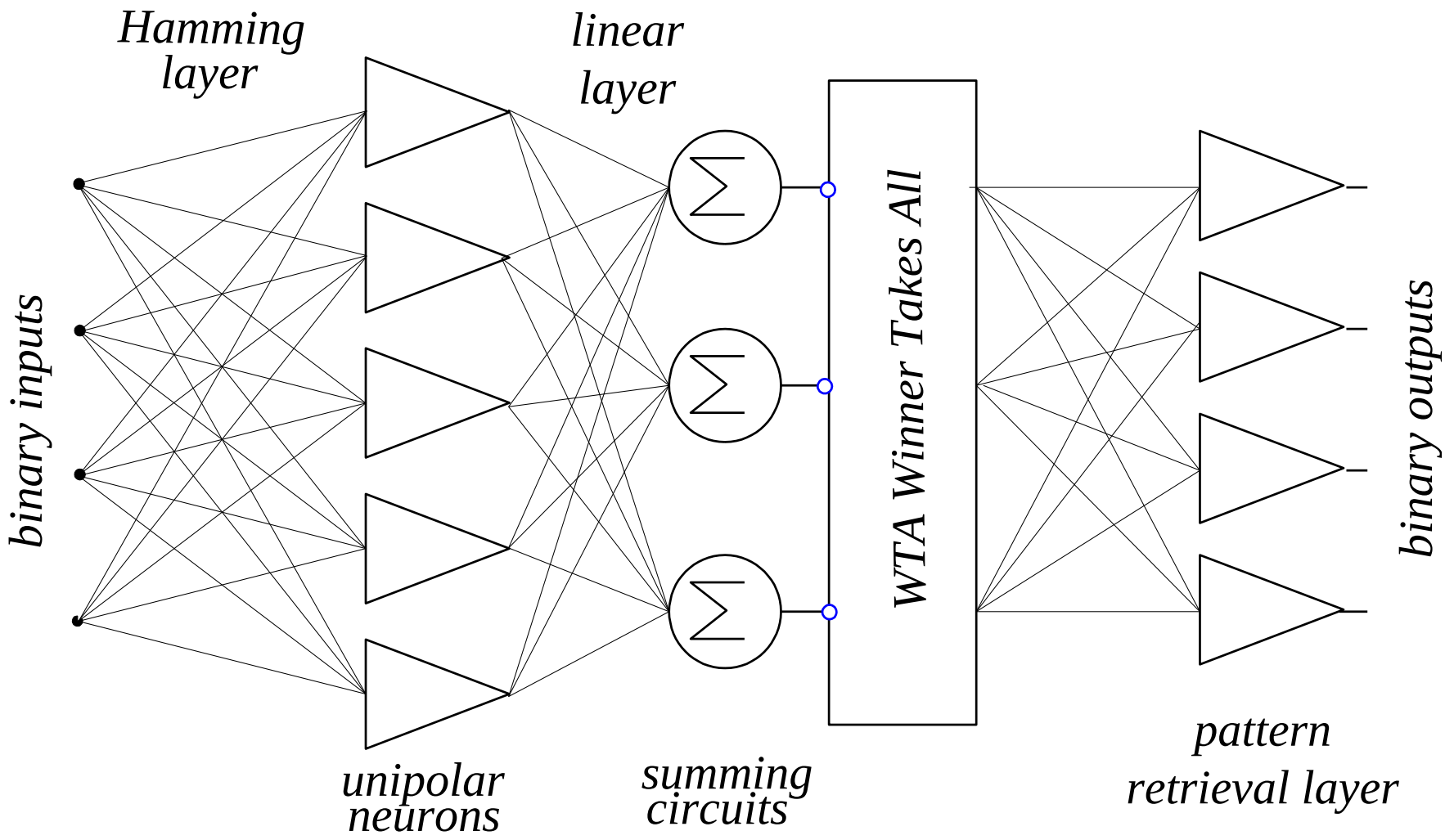











Hamming layer
linear layer
binary inputs
WTA Winner Takes All
binary outputs
unipolar neurons
summing circuits
pattern retrieval layer

The conclusion:

The system of computational intelligence can be smarter than humans

Is this new technological revolution?

- 150 years ago man power was replaced by machines (steam and electric)
- 20 years ago significant portion of man brain functions were replaced by computer *(calculations, administrative functions, voice and image recognitions etc)*
- We are still claiming that we are the most intelligent creatures in the universe, but for how much longer?

Artificial Intelligence or True Intelligence

Find clusters

Find number of clusters and its location in 4-dim. space

4	-3	4	7
4	-3	4	8
2	-5	3	6
4	-4	6	7
-4	5	6	-4
9	6	-3	4
2	-5	3	9
3	-3	5	6
5	-4	6	8
7	7	-2	3
-5	5	6	-3
9	6	-5	3
3	-5	6	8
3	-5	3	8
4	-6	5	7
-5	6	5	-4
3	-3	6	7
6	5	-5	4
-5	7	7	-3

-5	6	6	-3
7	4	-3	4
-3	6	5	-3
-5	6	8	-1
3	-5	6	7
-6	4	7	-1
3	-5	6	8
-4	4	5	-4
4	-5	5	7
-5	5	6	-2
9	7	-2	3
7	6	-4	2
8	5	-3	4
-5	4	7	-4
9	5	-4	2
8	6	-2	3
-3	5	7	-2
3	-6	3	7
4	-5	4	9

-4	4	5	-2
5	-3	5	7
4	-4	3	8
3	-6	5	8
4	-3	5	6
9	7	-3	2
3	-5	4	6
-4	6	6	-2
-5	4	7	-3
7	4	-2	4
9	6	-4	4
-6	7	5	-1
8	6	-3	3
-4	7	7	-2
8	5	-4	4
-4	5	8	-1
-3	4	7	-2
-4	5	6	-2

8	4	-4	3
6	6	-5	2
-5	6	7	-2
7	6	-5	4
3	-4	5	7
5	-4	6	9
4	-4	4	6
-5	6	7	-1
9	5	-4	2
-5	7	5	-1
-5	5	7	-2
8	5	-2	3
7	6	-5	5
-4	6	7	-3
6	5	-3	5
-5	6	5	-3
8	4	-2	4
-4	5	7	-4

Adding neurons as needed using minimum distance concept

much simpler and more efficient then ART

1. First pattern is applied and the first neuron is introduced

2. Next pattern is applied and then:

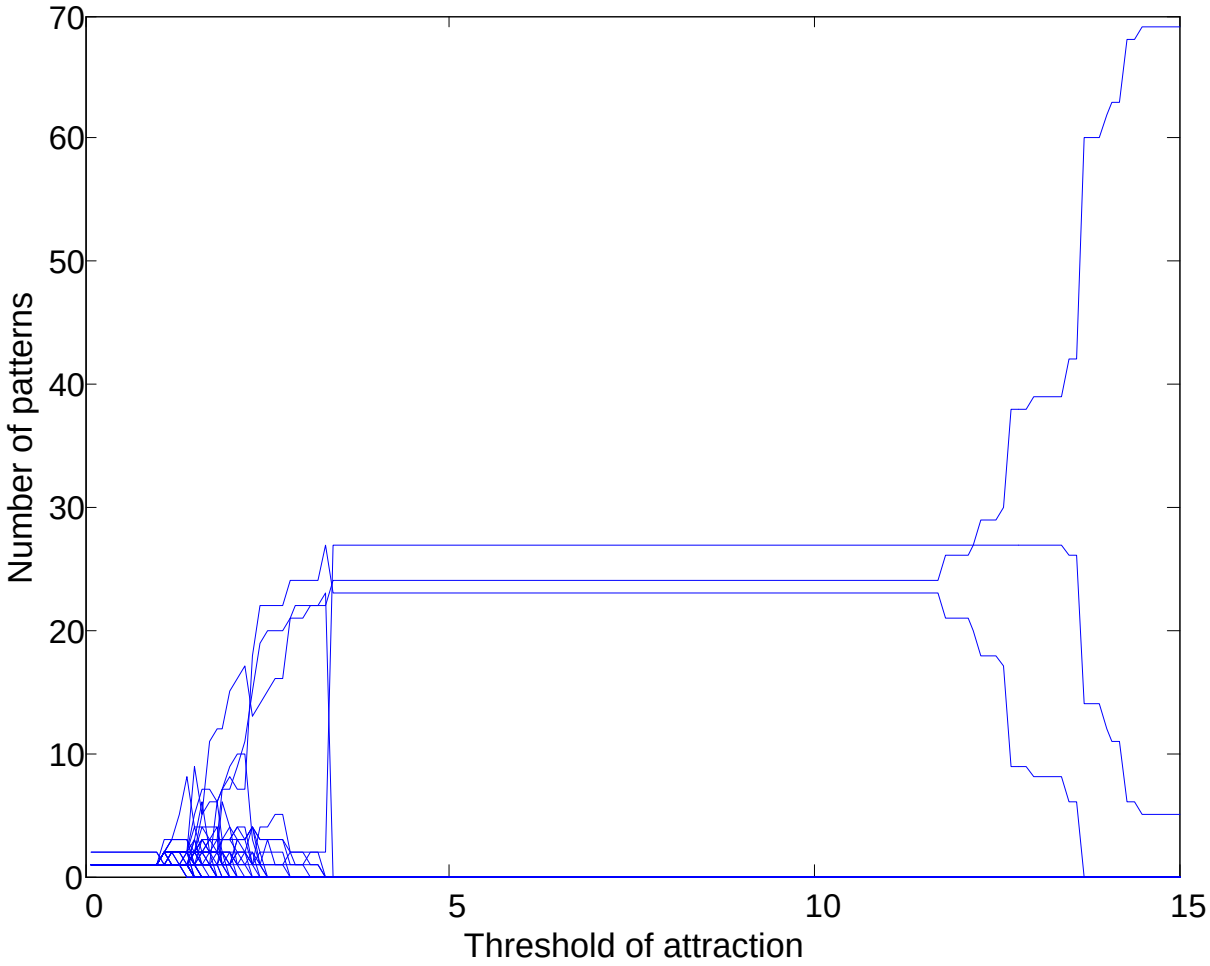
a) If distance form all existing clusters is larger then threshold then a new neuron is added

b) Else weights of the closest neuron are updated

$$\mathbf{W}_k = \frac{m\mathbf{W}_k + \alpha\mathbf{X}}{m+1}$$

where m is the number of previous patterns of a given set which were used to update this particular neuron and α is the learning constant

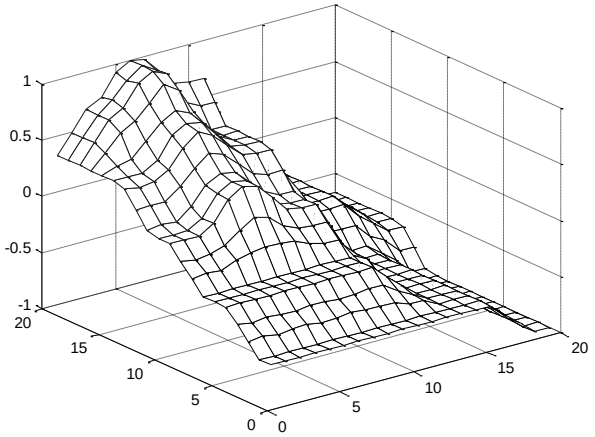
Adding neurons as needed using minimum distance concept

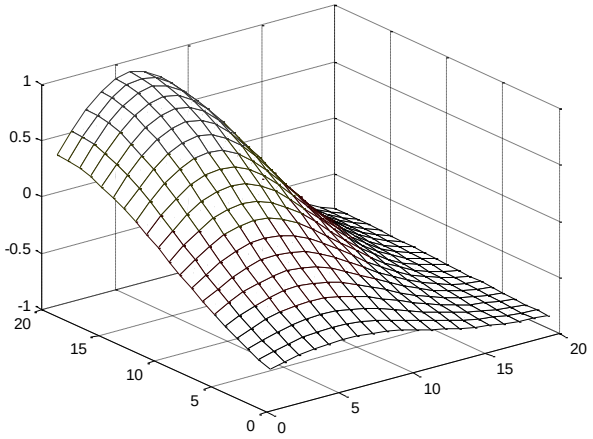


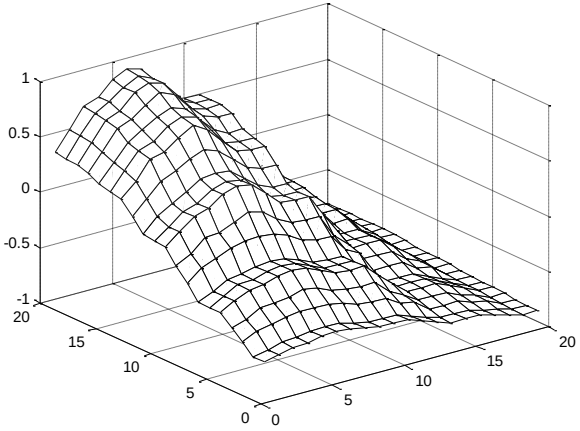
Limitations of fuzzy systems

There are two major ways to design fuzzy controllers:

1. Mamdani
2. Tagagi, Sugeno and Kun (TSK)



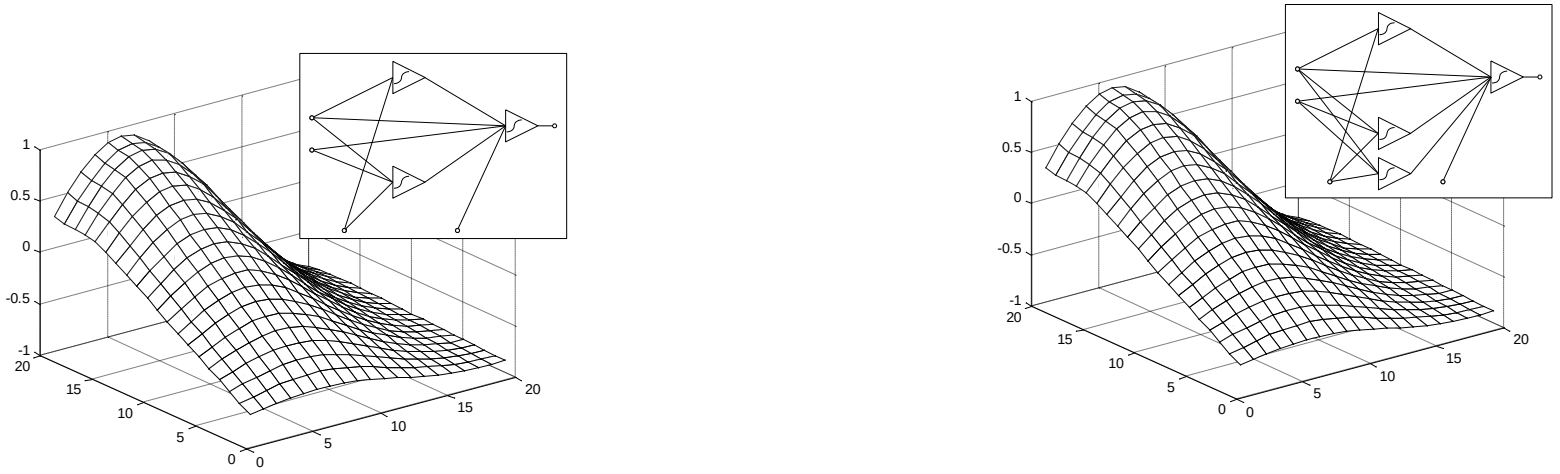




Control surface obtained with fuzzy controllers (a) required surface, (b) Mamdani controller with trapezoidal membership functions, (c) TSK controller with trapezoidal membership functions

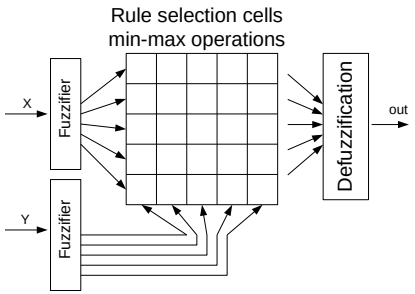
Number of inputs limited to 3 or 4

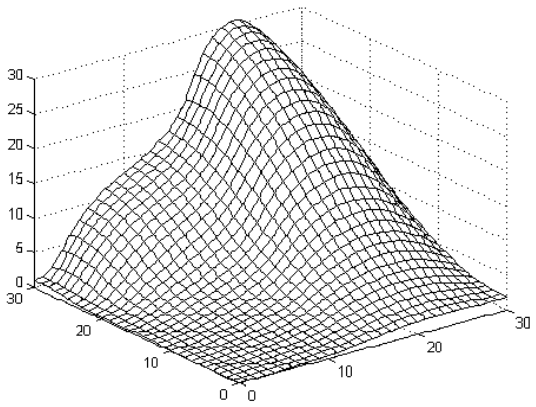
Limitations of neural networks

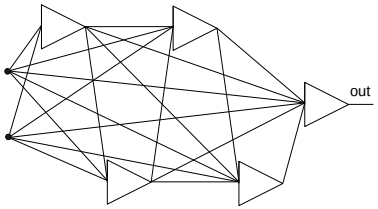


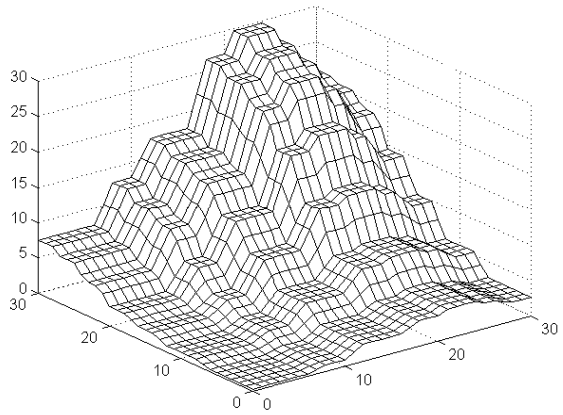
Control surfaces obtained with neural controller using (a) 3 neuron network, (b) 4 neuron network

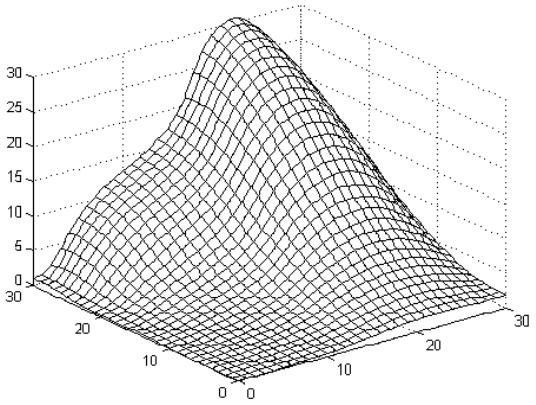
Comparison of fuzzy system and neural networks







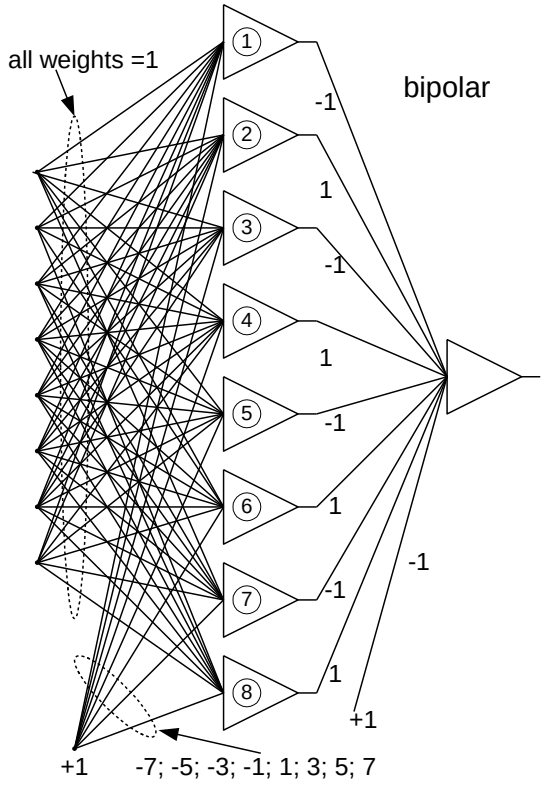




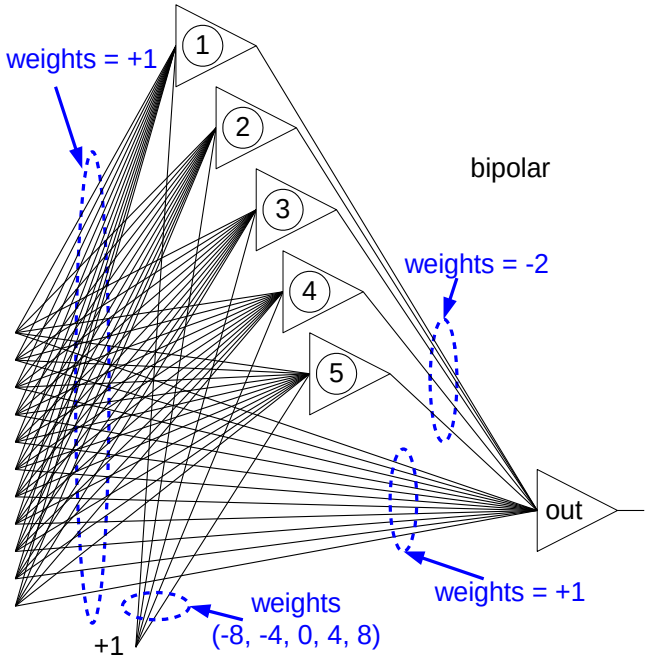
Control surfaces obtained with Motorola microcontroller HC11 using fuzzy approach with trapezoidal membership functions (7 functions per input) and Tagagi-Sugeno defuzzification

Control surfaces obtained with Motorola microcontroller HC11 using fuzzy approach with six neurons 2-1-1-1-1-1 architecture and Elliot activation function.

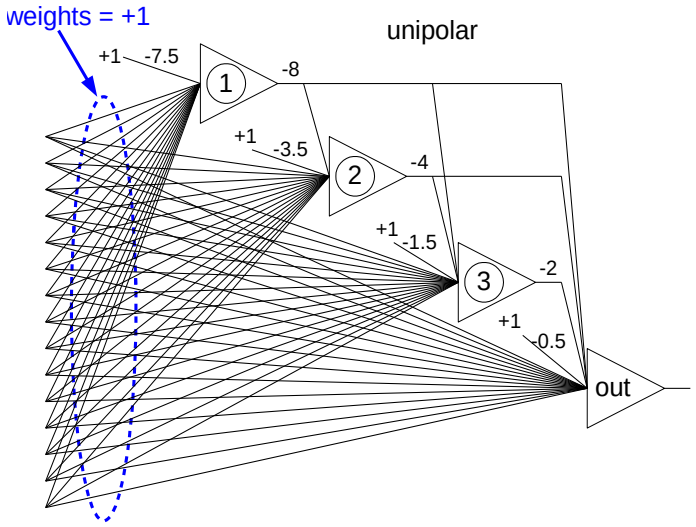
Various neural network architectures



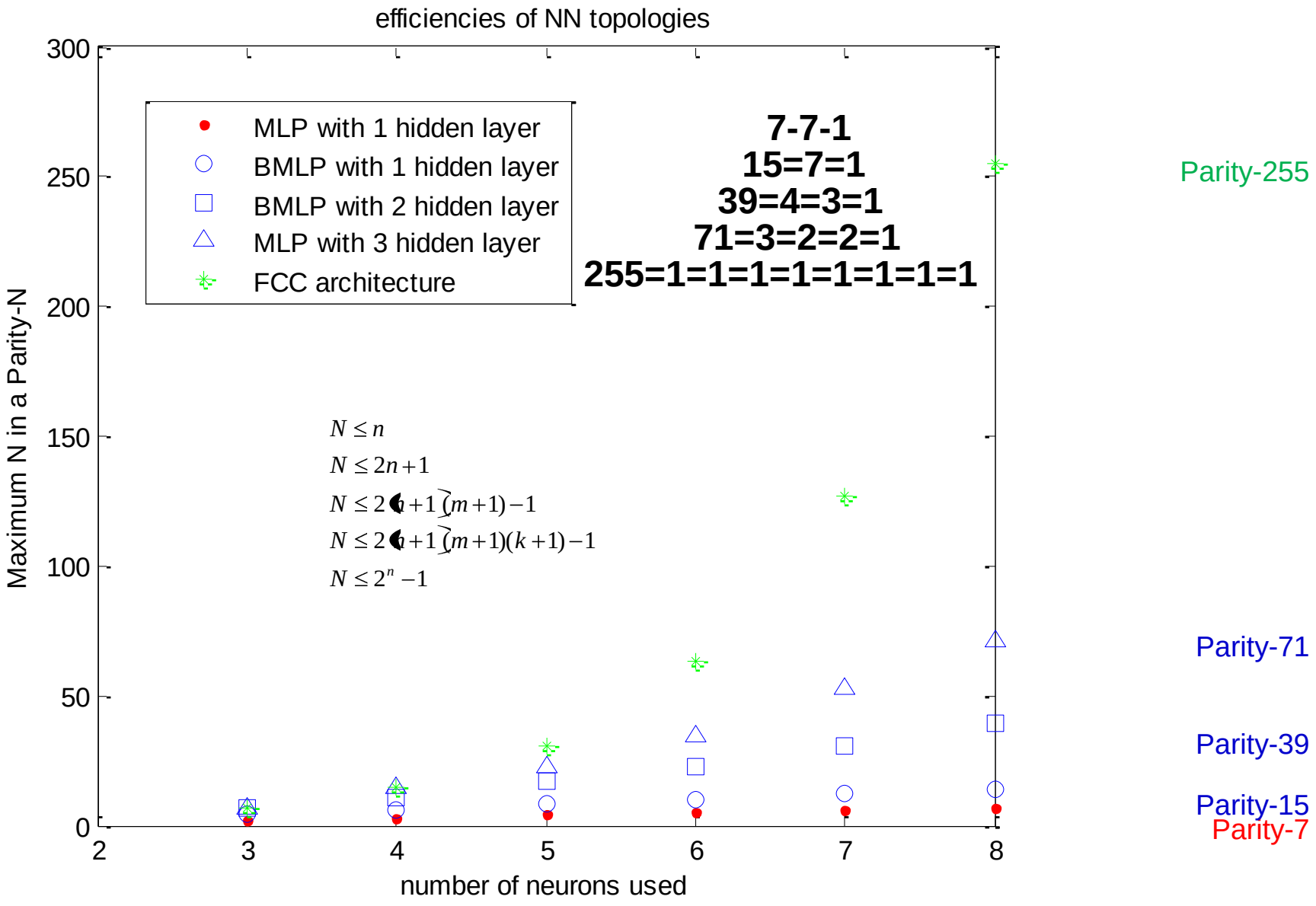
Layered bipolar neural network with one hidden layer for the **parity-8** problem.

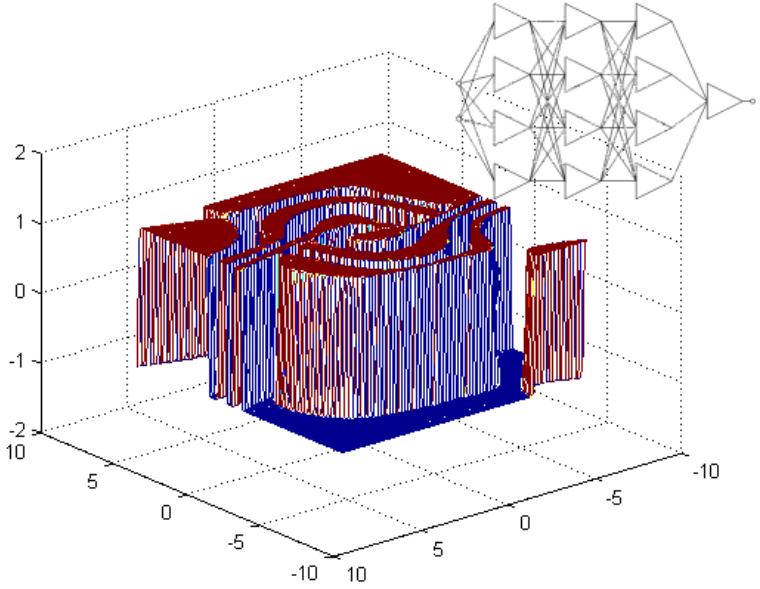


Parity-11 implemented in fully connected bipolar neural networks with five neurons in the hidden layer.

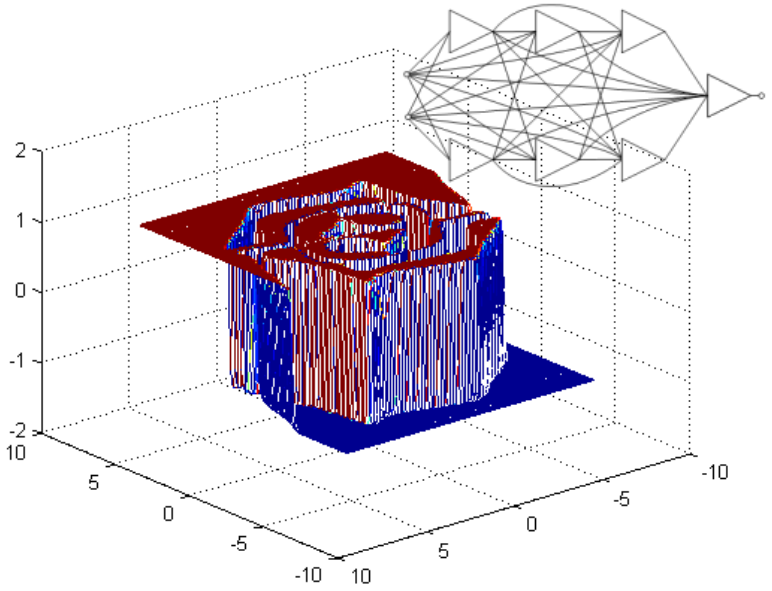


Parity-15 implemented with 4 neurons in one cascade

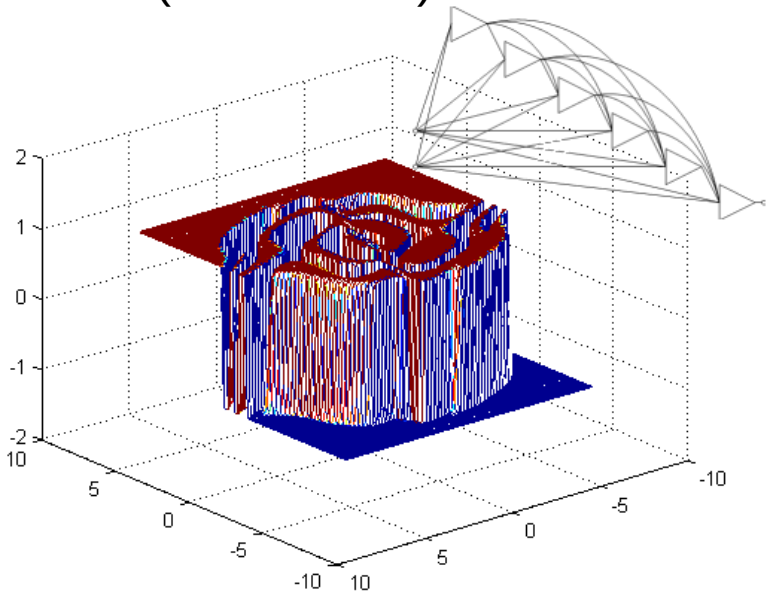

efficiencies of NN topologies
Maximum N in a Parity-N
number of neurons used
MLP with 1 hidden layer
BMLP with 1 hidden layer
BMLP with 2 hidden layer
MLP with 3 hidden layer
FCC architecture
7-7-1
15=7=1
39=4=3=1
71=3=2=2=1
255=1=1=1=1=1=1=1=1
$N \le n$
$N \le 2n+1$
$N \le 2\left(\frac{n}{2}+1\right)(m+1)-1$
$N \le 2\left(\frac{n}{2}+1\right)(m+1)(k+1)-1$
$N \le 2^n - 1$
Parity-255
Parity-71
Parity-39
Parity-15
Parity-7



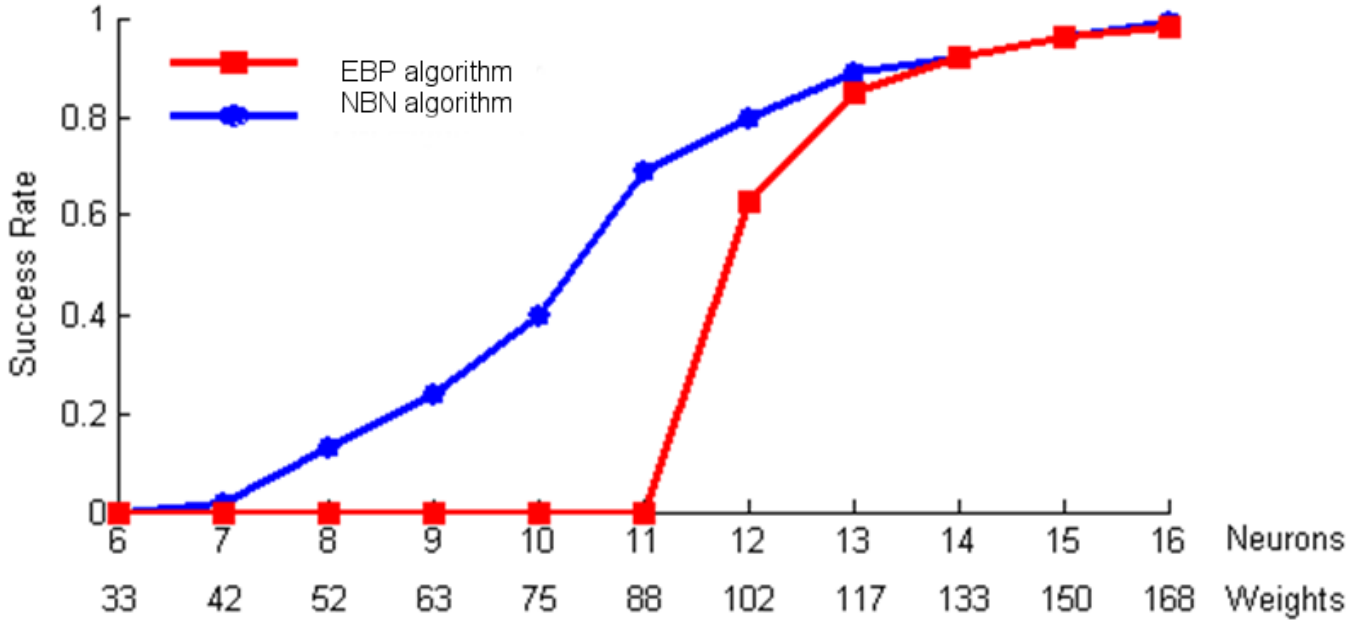
Solution of the two spiral problem using MLP architecture with 13 neurons (2-4-4-4-1).



Solution of the two spiral problem using BMLP architecture with 7 neurons (2=2=2=2=1).



Solution of the two spiral problem using FCC architecture with 6 neurons (2=1=1=1=1=1=1)

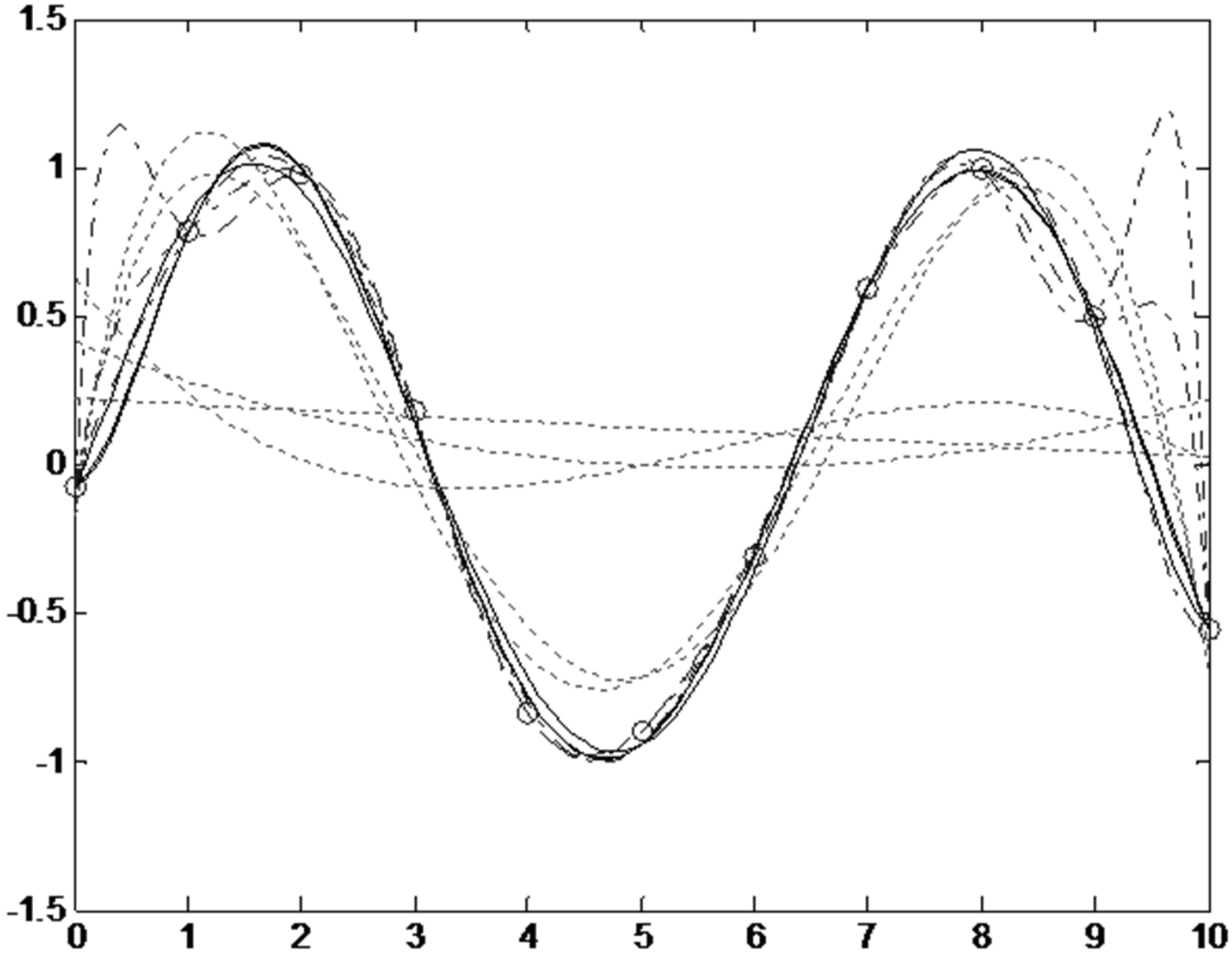


Training results of two-spiral problem

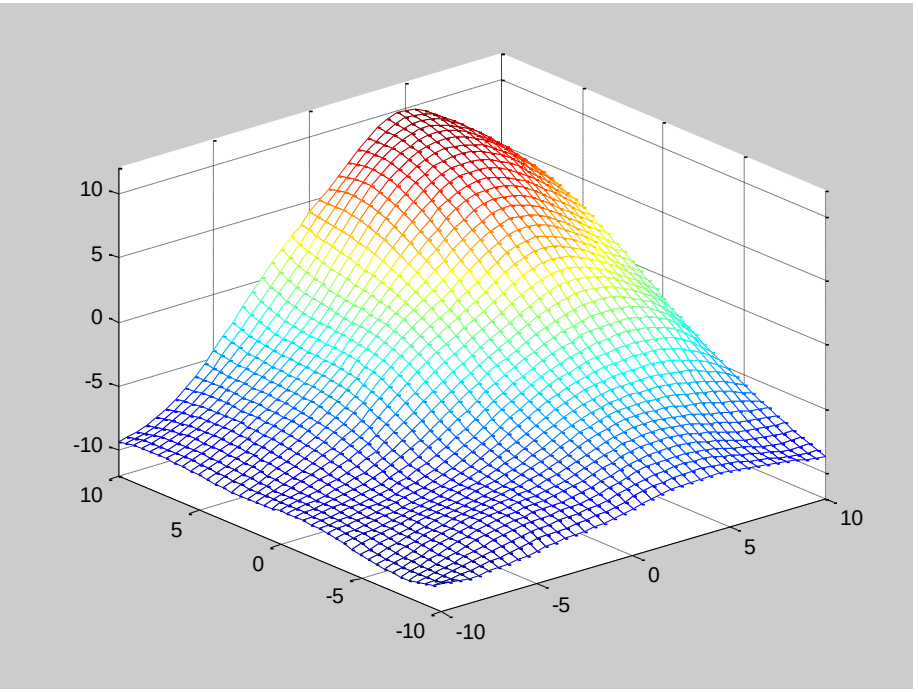
Neurons	Success rate		Average number of iterations		Average time (s)	
	EBP	NBN	EBP	NBN	EBP	NBN
8	0%	13%	Failing	287.7	Failing	0.88
9	0%	24%	Failing	261.4	Failing	0.98
10	0%	40%	Failing	243.9	Failing	1.57
11	0%	69%	Failing	231.8	Failing	1.62
12	63%	80%	410,254	175.1	633.91	1.70
13	85%	89%	335,531	159.7	620.30	2.09
14	92%	92%	266,237	137.3	605.32	2.40

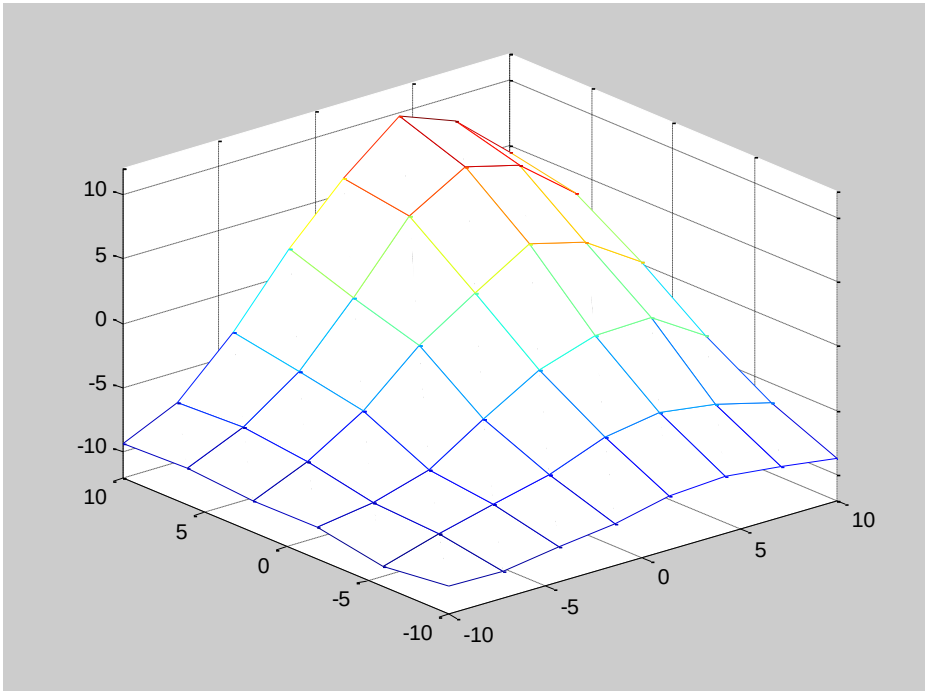
For EBP algorithm, learning constant is 0.005 (largest possible to avoid oscillation) and momentum is 0.5; maximum iteration is 1,000,000 for EBP algorithm and 1,000 for LM algorithm; desired error=0.01; all neurons are in fully connected cascade networks; there are 100 trials for each case.

Polynomial approximations



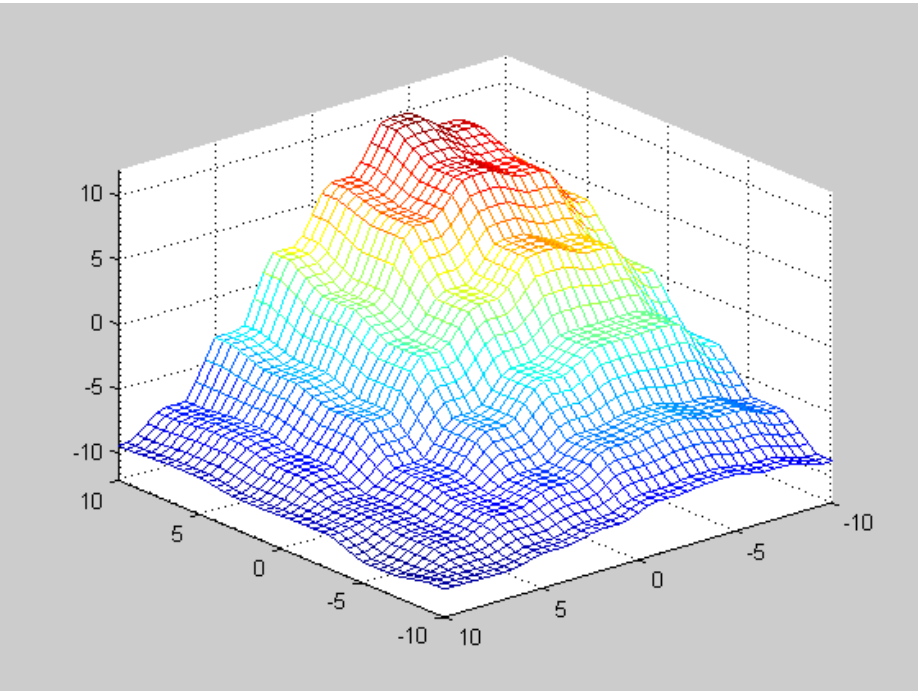
Performance of Fuzzy Systems and Neural Networks

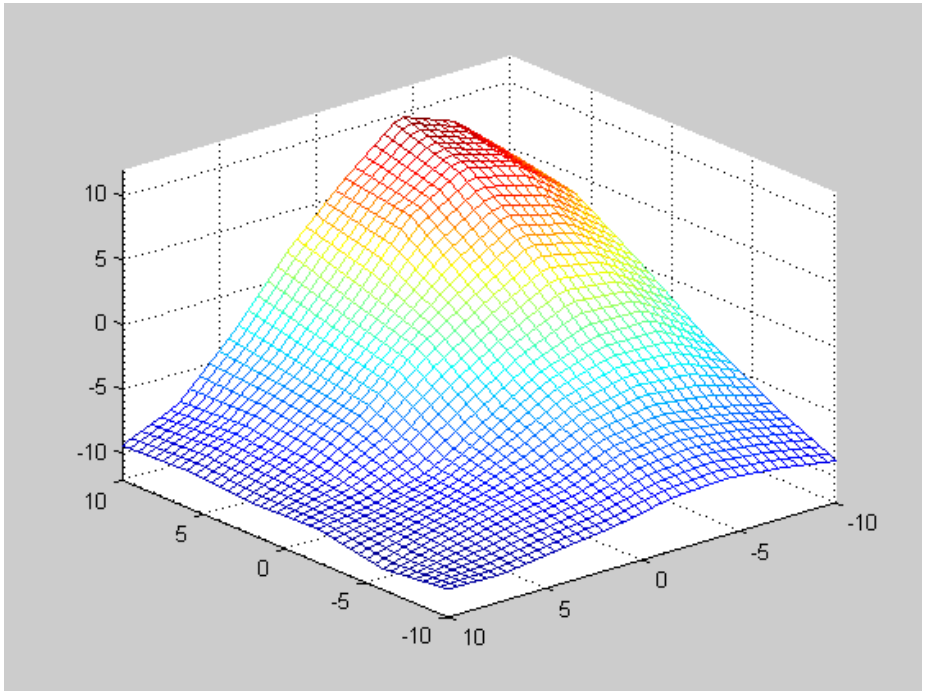




Control surface of TSK fuzzy controller (a) required control surface (b) 8*6=48 defuzzyfication rules

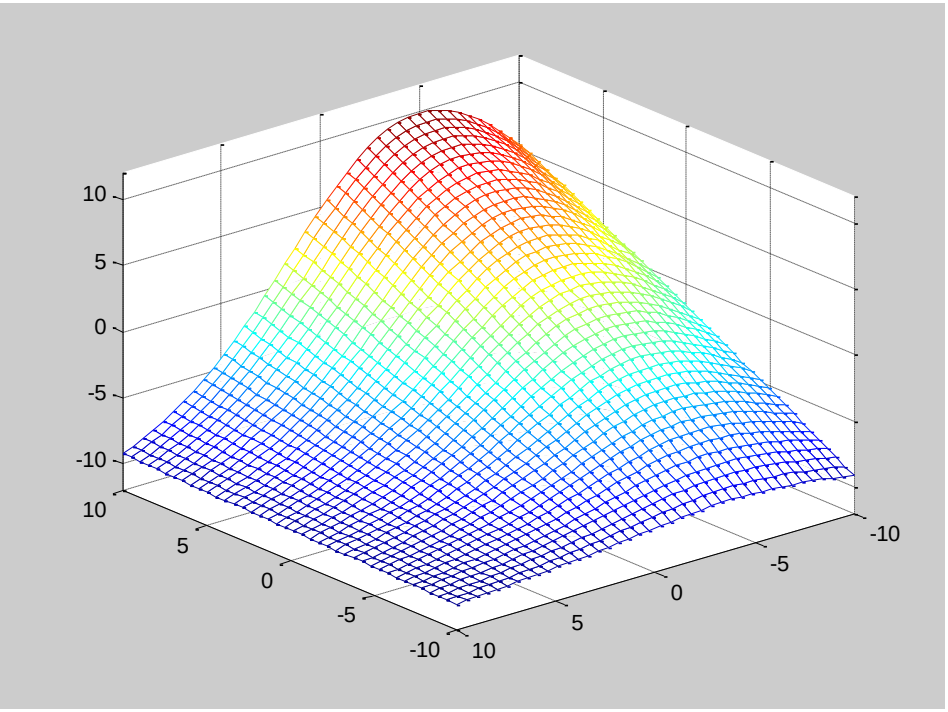
Performance of Fuzzy Systems

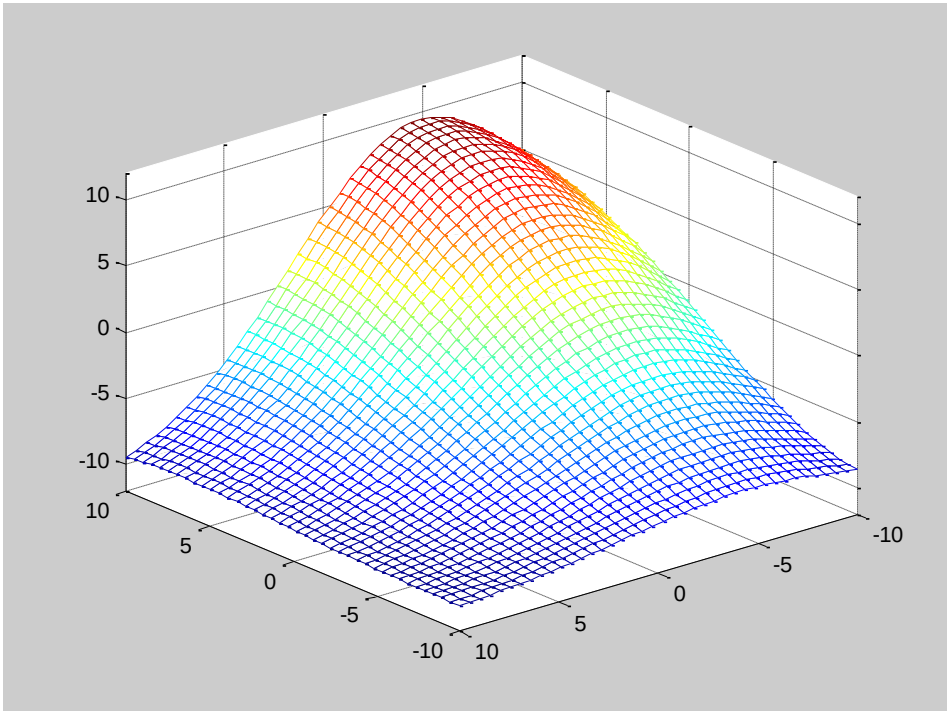




Control surface of TSK fuzzy controller with equally spaced membership function 8 in x-direction and 6 in y-direction (a) trapezoidal membership functions (b) triangular membership

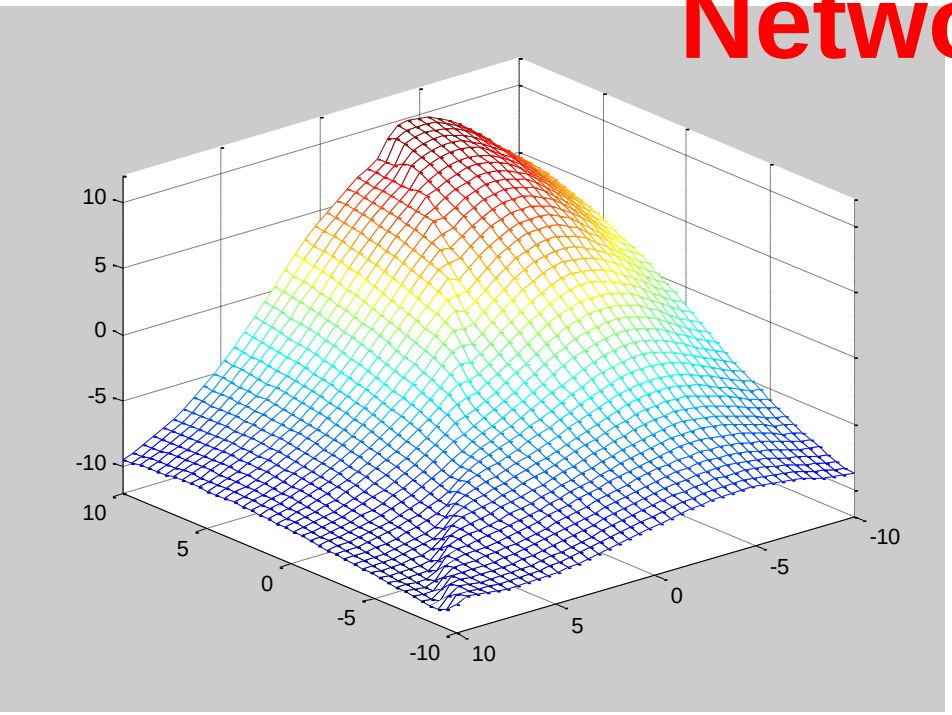
Performance of Neural Networks

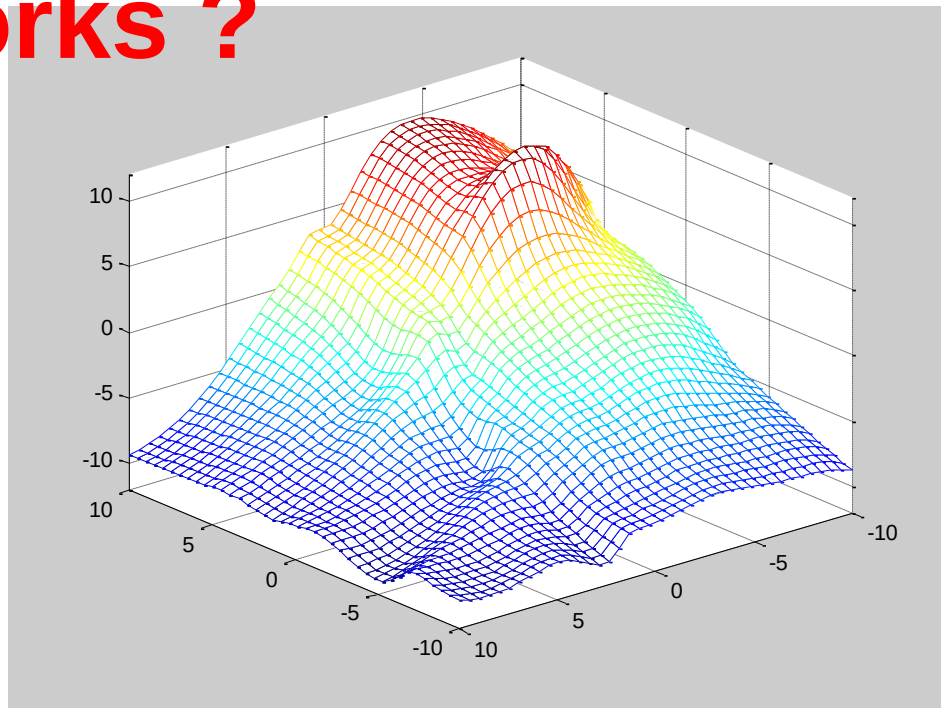




Control surface obtained with neural networks (a) 3 neurons in cascade (12 weights) Training Error=0.21049 (b) 4 neurons in cascade (18 weights) Training Error=0.049061

What is wrong with Neural Networks ?

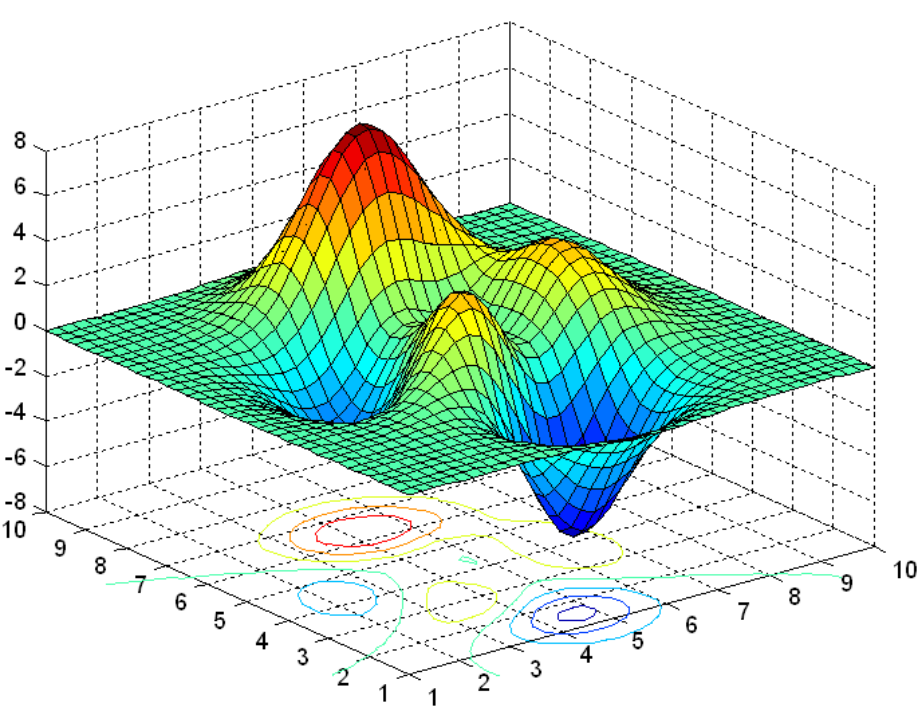


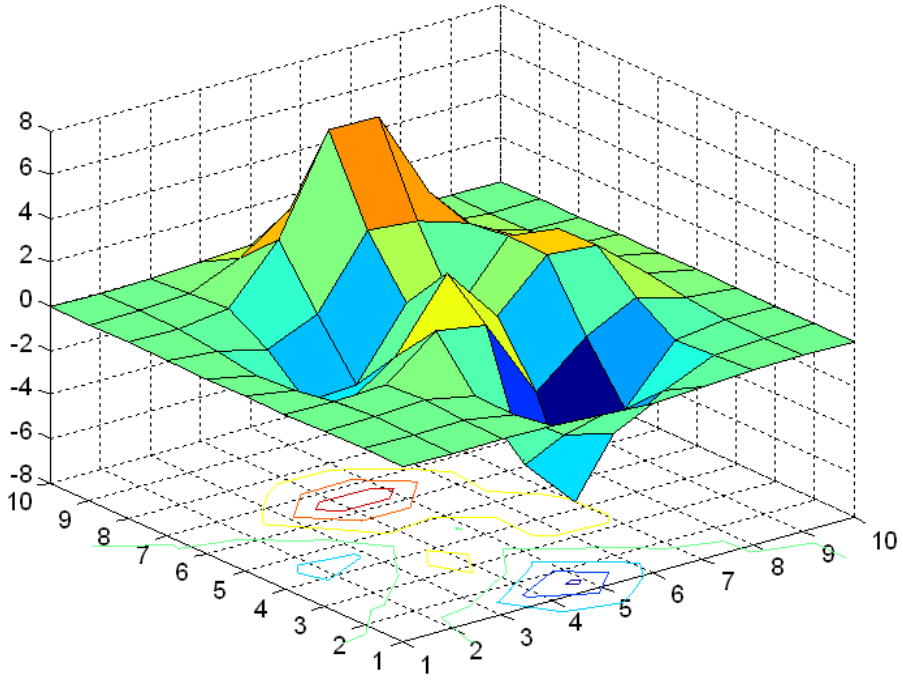


Control surface obtained with neural networks (a) 5 neurons in cascade (25 weights) Training Error=0.023973 (b) 8 neurons in cascade (52 weights) Training Error=1.118e-005

EBP is not able to train optimal architectures

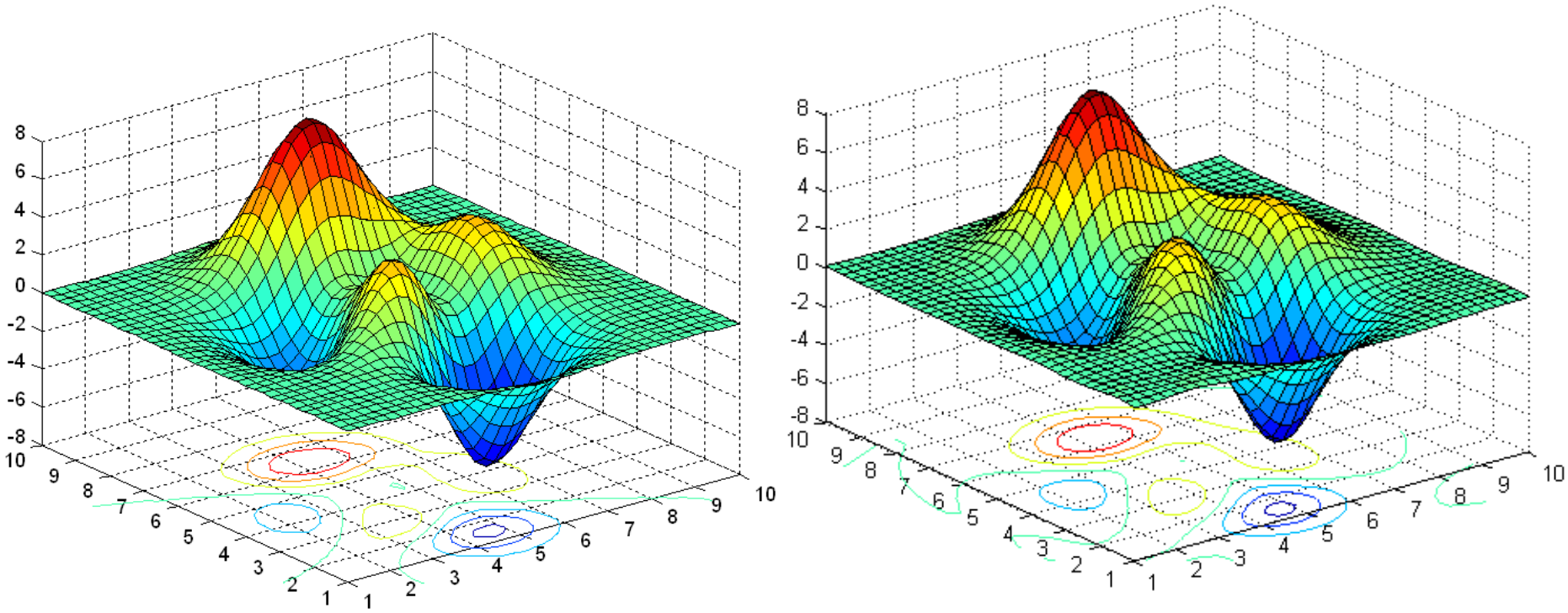
Other example





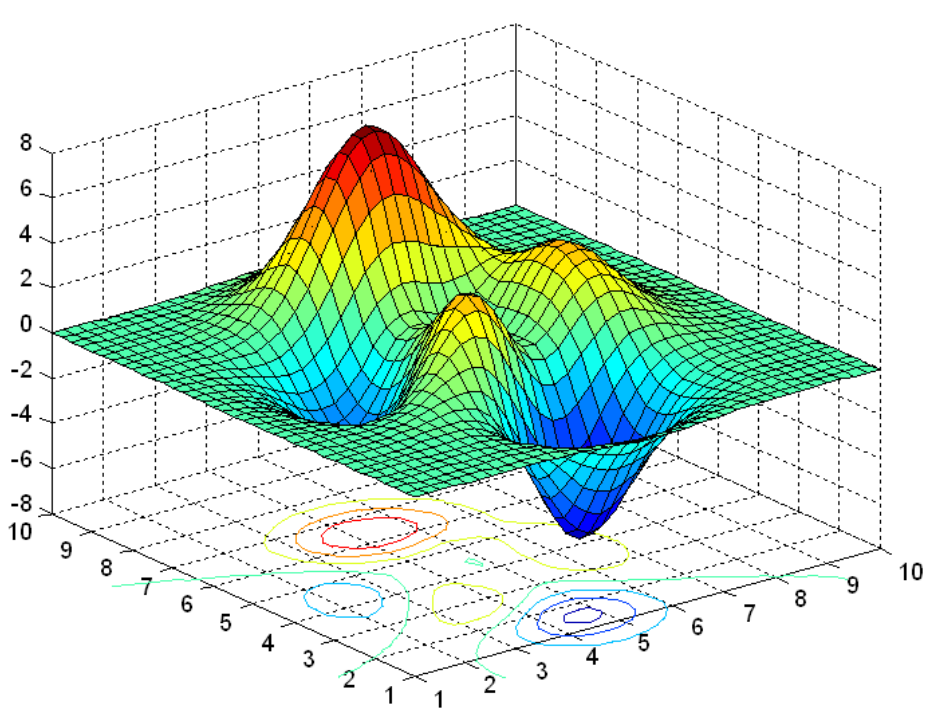
Surface matching problem: (a) Required 2-D surface with *37×37=1,369* points, used for verification; (b) *10×10=100* training patterns extracted in equal space from (a), used for training.

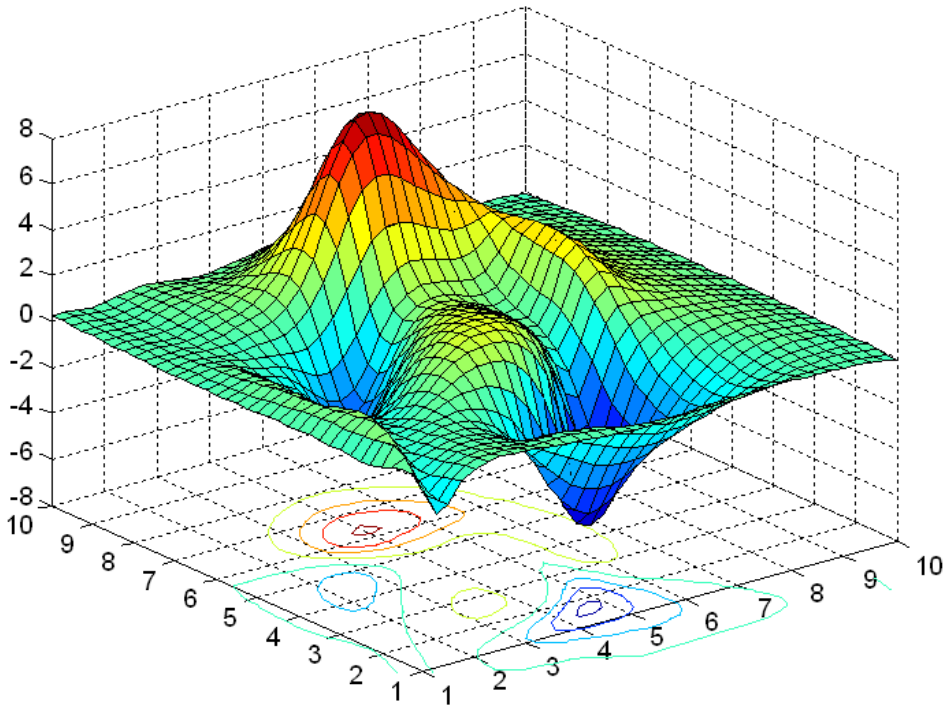
NBN – 8 neurons



The best training result in 100 trials, using LM algorithm, **8 neurons** in FCC network (52 weights); maximum training iteration is 1,000; SSE_{Train}=0.0044, SSE_{Verify}=0.0080 and training time=0.37 s.

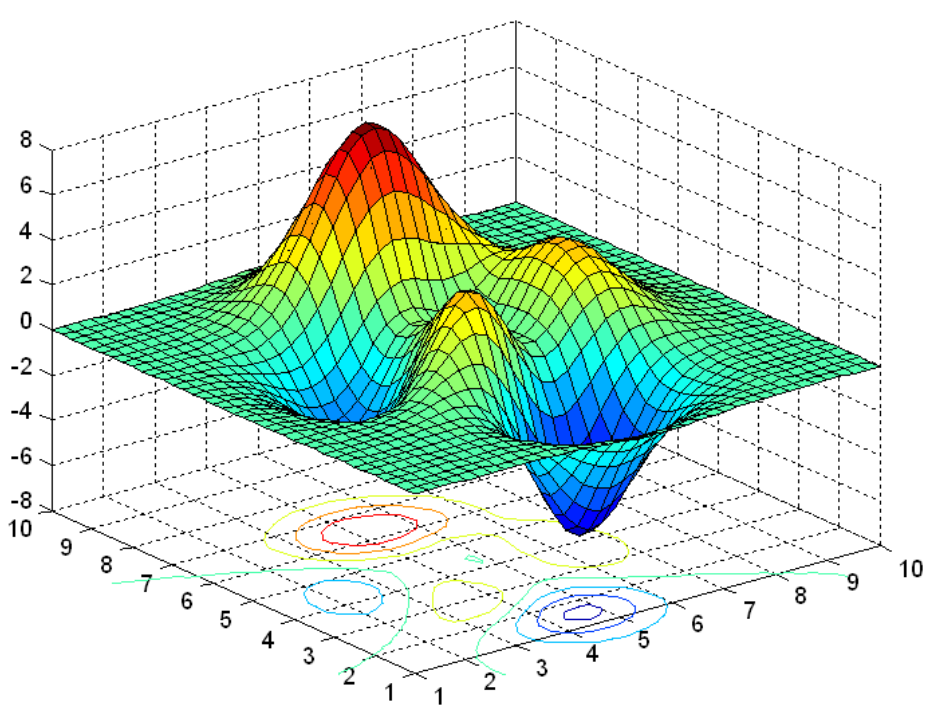
EBP – 8 neurons

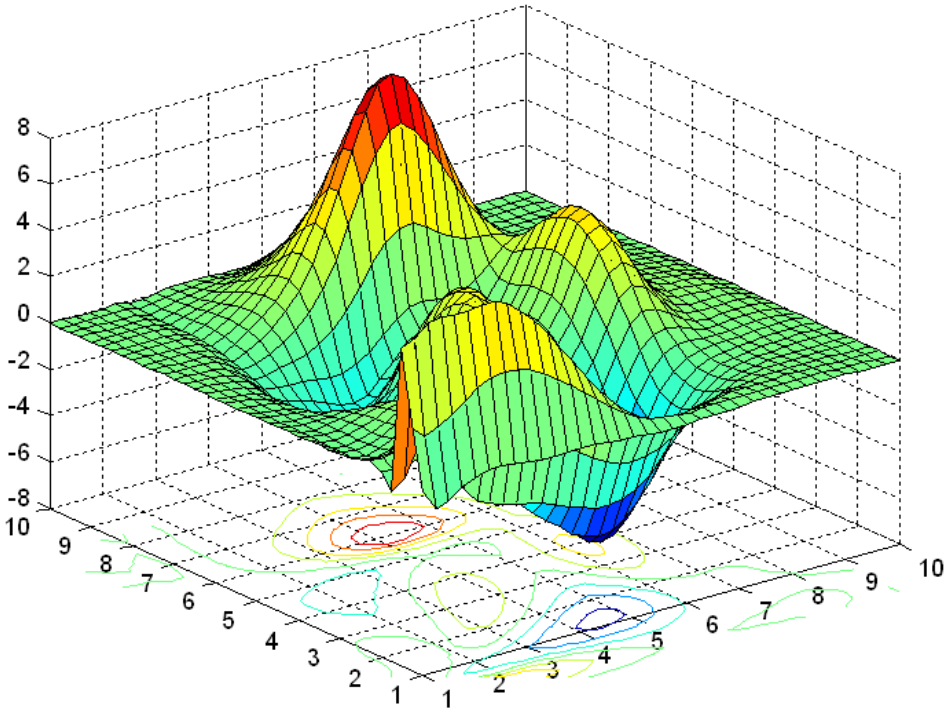




The best training result in **100** trials, using EBP algorithm, 8 neurons in FCC network (52 weights); maximum training iteration is **1,000,000**; SSE_{Train}=0.0764, SSE_{Verify}=0.1271 and training time=579.98 s.

EBP – 13 neurons

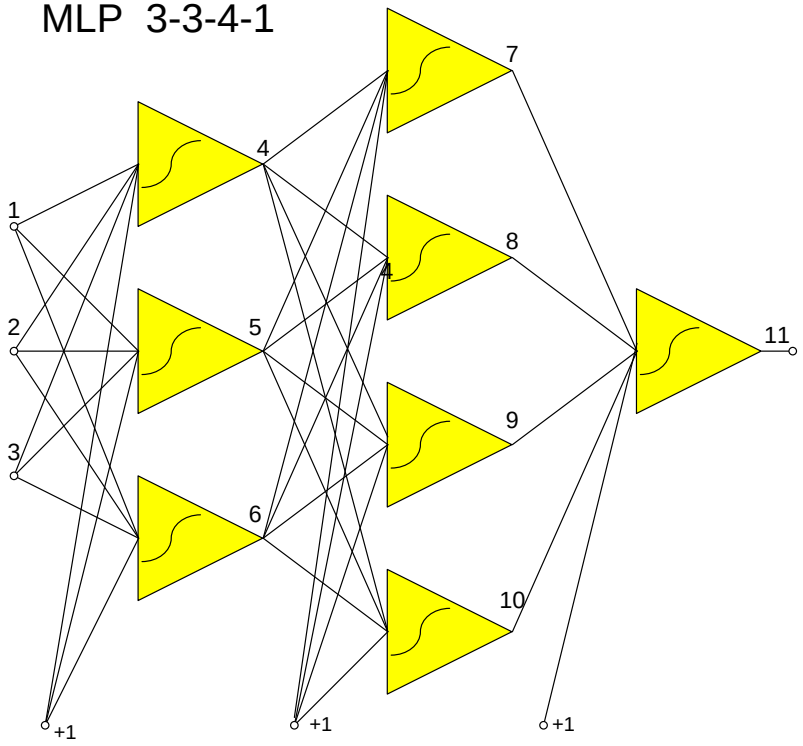




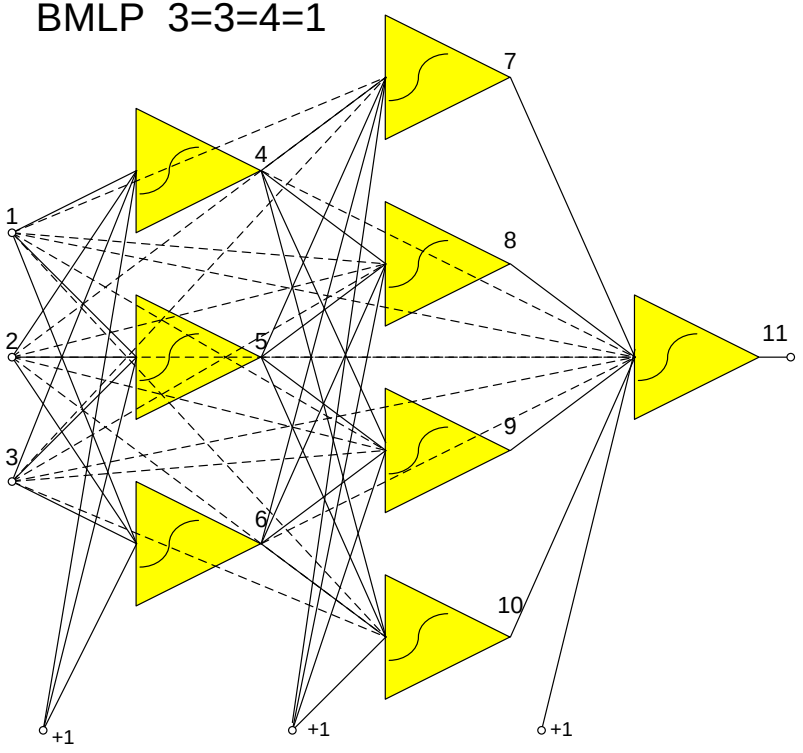
The best training result in 100 trials, using EBP algorithm, 13 neurons in FCC network (117 weights); maximum training iteration is **1,000,000**; **SSE_{Train}=0.0018**, SSE_{Verify}=0.4909 and training time=635.72 s.

Best neural network architectures

MLP 3-3-4-1



BMLP 3=3=4=1



Most software can train only MLP

Supervised learning rules for single neuron

$$\Delta \mathbf{w}_i = c\,\delta\,\mathbf{x}$$

correlation rule (supervised): $\delta = d$

perceptron fixed rule: $\delta = d - o$

perceptron adjustable rule - as above but the learning constant is modified to:

$$\alpha^* = \alpha\,\lambda\frac{\mathbf{x}\mathbf{w}^T}{\mathbf{x}\mathbf{x}^T} = \alpha\,\lambda\frac{net}{\|\mathbf{x}\|^2}$$

LMS (Widrow-Hoff) rule: $\delta = d - net$

delta rule: $\delta = (d - o)f'$

pseudoinverse rule (the same as LMS): $\mathbf{w} = (\mathbf{x}^T\mathbf{x})^{-1}\mathbf{x}^T d$

Levenberg-Marquardt Algorithm (LM)

Newton method:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mathbf{A}_k^{-1}\mathbf{g}$$

Assumptions:

$$\mathbf{A} \approx 2\mathbf{J}^T\mathbf{J} \quad \text{and} \quad \mathbf{g} = 2\mathbf{J}^T\mathbf{e}$$

where **J** is Jacobian and **e** is error vector

Gauss-Newton method:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \left(2\mathbf{J}_k^T\mathbf{J}_k\right)^{-1} 2\mathbf{J}_k^T\mathbf{e}$$

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \left(\mathbf{J}_k^T\mathbf{J}_k\right)^{-1}\mathbf{J}_k^T\mathbf{e}$$

Levenberg - Marquardt method:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \left(\mathbf{J}_k^T\mathbf{J}_k + \mu\mathbf{I}\right)^{-1}\mathbf{J}_k^T\mathbf{e}$$

Levenberg-Marquardt Algorithm (LM)

Very powerful and very fast

Can train only MLP architectures so optimal and most powerful architectures cannot be trained

Size of Jacobian proportional to number of patterns so only relatively small problems can be solved

Neuron by Neuron Algorithm (NBN)

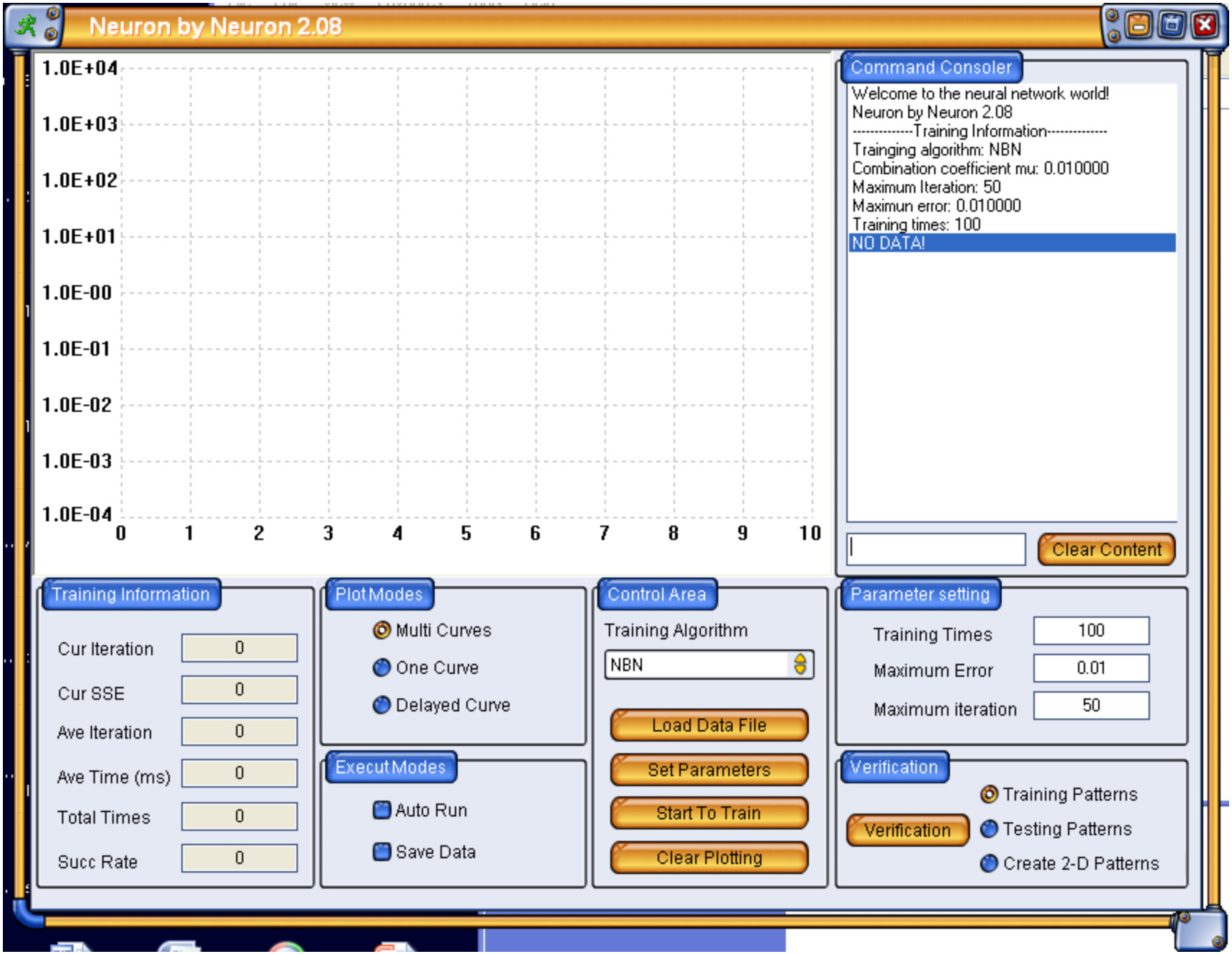
Very powerful and very fast

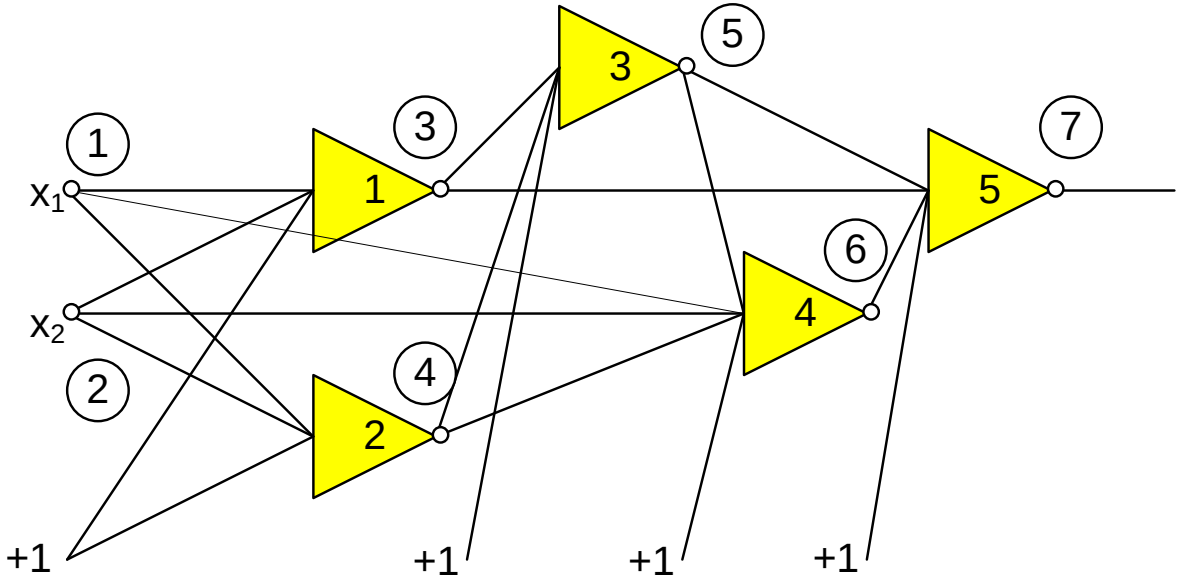
Can train any NN architectures so any architectures can be trained (including most powerful ones)

Jacobian is not calculated and stored so problems with basically unlimited number of patterns can be effectively trained

Easy way of adding (or removing training patterns without a necessity of retraining of entire set

Only forward pass in the contrast to most algorithms where both forward and backward computations are needed

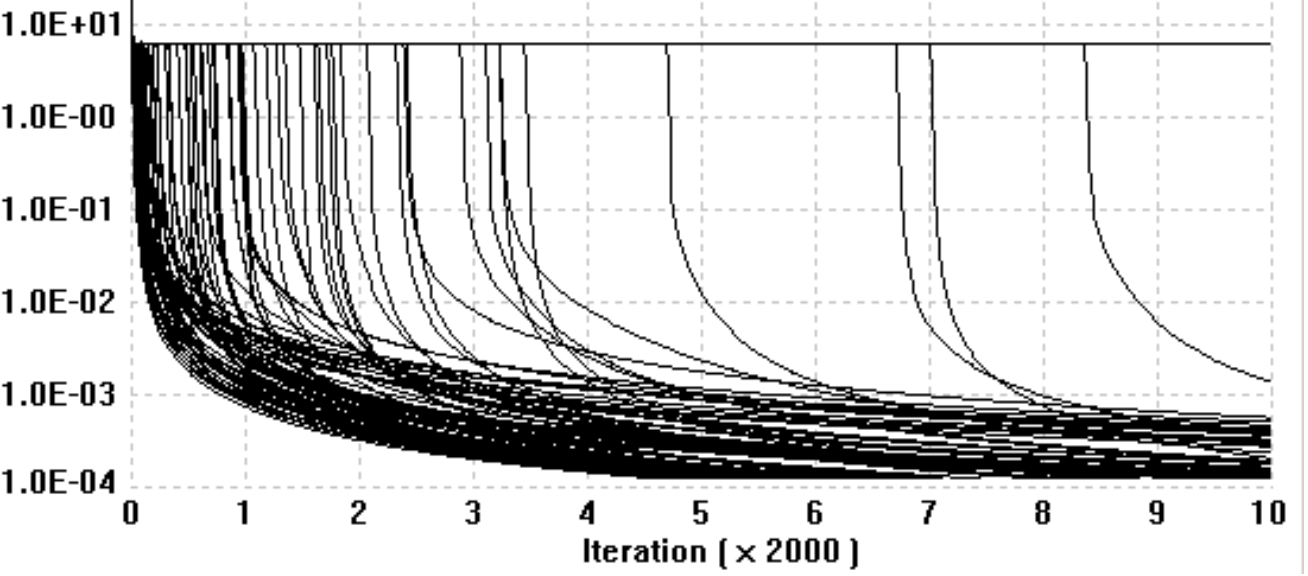

Neuron by Neuron 2.08
1.0E+04
1.0E+03
1.0E+02
1.0E+01
1.0E-00
1.0E-01
1.0E-02
1.0E-03
1.0E-04
0
1
2
3
4
5
6
7
8
9
10
Command Consoler
Welcome to the neural network world!
Neuron by Neuron 2.08
----------Training Information----------
Trainging algorithm: NBN
Combination coefficient mu: 0.010000
Maximum Iteration: 50
Maximun error: 0.010000
Training times: 100
NO DATA!
Clear Content
Training Information
Cur Iteration
0
Cur SSE
0
Ave Iteration
0
Ave Time (ms)
0
Total Times
0
Succ Rate
0
Plot Modes
Multi Curves
One Curve
Delayed Curve
Execut Modes
Auto Run
Save Data
Control Area
Training Algorithm
NBN
Load Data File
Set Parameters
Start To Train
Clear Plotting
Parameter setting
Training Times
100
Maximum Error
0.01
Maximum iteration
50
Verification
Verification
Training Patterns
Testing Patterns
Create 2-D Patterns



n_1 [model] 3 1 2
n_2 [model] 4 1 2
n_3 [model] 5 3 4
n_4 [model] 6 1 2 4 5
n_5 [model] 7 3 5 6

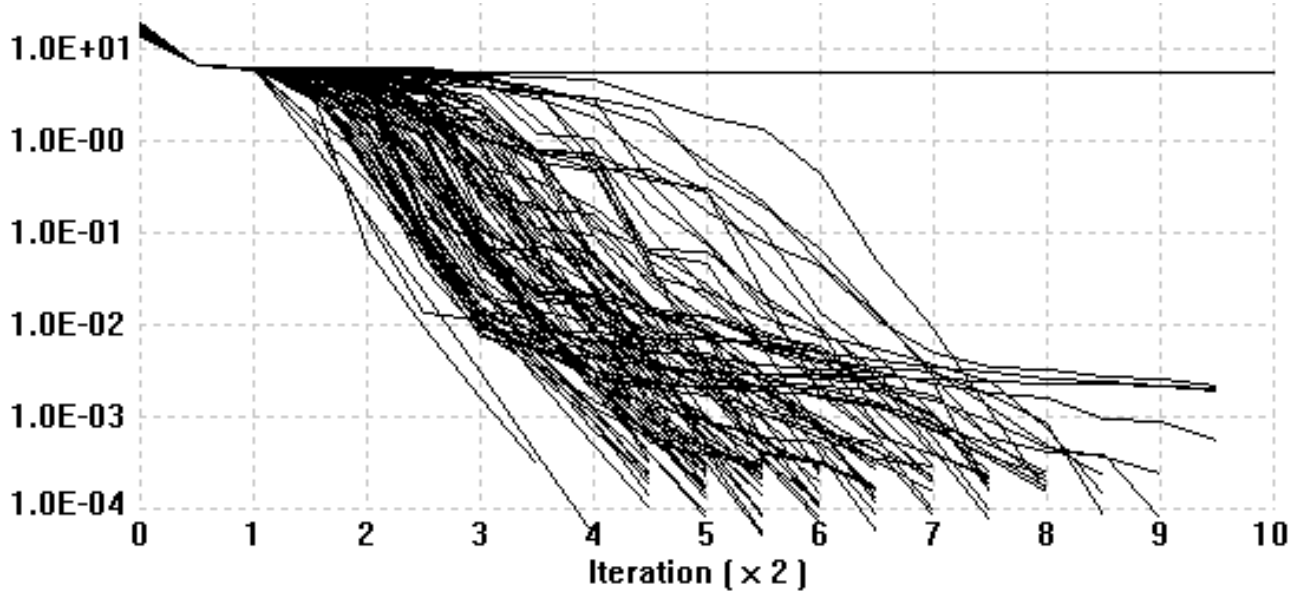
```
for all patterns (np)
% Forward computation
  for all neurons (nn)
    for all weights of the neuron (nx)
      calculate net;
    end;
    calculate neuron output;
    calculate neuron slope;
  end;
  for all outputs (no)
    calculate error;
```

```
%Backward computation
    initial delta as slope;
    for all neurons starting from output neurons (nn)
      for the weights connected to other neurons (ny)
        multiply delta through weights
        sum the backpropagated delta at proper nodes
      end;
      multiply delta by slope (for hidden neurons);
    end;
    related Jacobian row computation;
  end;
end;
```



Number of iterations	7956
Training time	3297ms
Success rate	46%

Sum of squared errors as a function of number of iterations for the Parity-4 problem using EBP algorithm, and 100 runs



Number of iterations	17
Training time	15ms
Success rate	69%

Result of parity-4 training using NBN algorithm with 4-3-3-1 architecture, and 100 runs

Conclusions

✓**Researchers are using wrong architectures: MLP instead of ACN**

✓**Researchers are using excessive number of neurons**

✓**First order algorithm such as EBP is not able to train optimal networks**

✓**Popular second order algorithm such as LM can train only MLP networks**

The newest version of NBN software can be downloaded from

http://www.eng.auburn.edu/~wilambm/nnt/